

# ECHO 9.0 Data Partner's Guide

For ECHO Version 9.0

NASA

April 25, 2007

1.0 Edition

**This page is intentionally left blank.**

# Table of Contents

Table of Contents .....	i
Table of Code Listings .....	vii
Preface .....	ix
Conventions .....	ix
Chapter 1: Before You Begin .....	1
Tasks That You Will Perform as a Data Partner.....	1
Skills You Will Need as a Data Partner.....	2
ECHO Concept and Design .....	2
ECHO as a Spatially Enabled Metadata Search and Order System.....	3
Security .....	4
Supported Platforms.....	4
ECHO Capability and Functionality .....	4
Benefits to Data Partners .....	5
Chapter 2: The Basics .....	7
Locating the ECHO Web Services .....	7
ECHO Globally Unique Identifiers (GUIDS).....	8
ECHO Entities .....	8
Users .....	8
Roles .....	8
Queries (Used Primarily by Client Partners) .....	9
Catalog Items .....	9
Orders.....	9
Order Options.....	10
Authenticators .....	11
Groups.....	11
Conditions .....	11
Rules .....	11
Extended Services .....	12

Chapter 3: Ingest .....	13
Ingest Overview .....	13
Automated Operational Ingest .....	14
Rules for Automated Ingest .....	14
Notification of Metadata Transmission .....	14
Considerations for Ingest Metadata .....	14
Additional Attributes .....	14
Metadata Mapping/Ingest Process .....	14
Creating ECHO-Compatible XML Files .....	15
ECHO Collections .....	16
Basic Collection Information.....	16
Spatial Data.....	17
Temporal Data .....	19
Sources and Sensors.....	20
Keywords .....	23
Additional Attributes .....	23
Other Descriptive Information.....	24
Restriction Flag for a Collection.....	24
ECHO Granules .....	25
Granule/Collection Association .....	25
Basic Granule Information.....	26
Sources and Sensors.....	28
Additional Attributes .....	30
Measured Parameters .....	31
Orbital Information .....	31
Other Descriptive Information.....	32
Restriction Flag for a Granule.....	36
Spatial Representations, Coordinates and Projections.....	36
Two-Dimensional Coordinate System.....	37
Geometry Representations .....	37
Coordinate System .....	37
Data Types and Representation .....	39

Invalid Spatial Representation .....	49
Polygon Points in Counter-Clockwise Order.....	49
Twisted Polygon .....	50
Hole Crosses over Outer Ring .....	51
Polygon Crosses International Date Line .....	52
Overlapping Polygons.....	54
Inappropriate Data .....	56
Incorrectly Defined Spatial Coverage.....	59
Latitude/Longitude Range and Tolerance.....	60
Online Data Access URL and Online Resources URL.....	60
Ingest Reporting.....	60
New Items vs. Update Items .....	60
Metadata Ingest DTD.....	62
Delete Items .....	63
Data Not Included in the DTD.....	63
Values for the Qualified Tag for Updates, Deletes, and Inserts .....	63
Examples.....	67
Browse Image Files and Browse Metadata.....	70
Browse Insert, Update, Replace and Delete.....	72
Error Messages.....	77
Chapter 4: Validating Your Metadata.....	85
View Dataset Information.....	85
Parameters:.....	85
Returns:.....	85
Chapter 5: Controlling Access to Your Metadata.....	87
Managing Data Access Rules .....	87
Data Access Rules.....	87
Chapter 6: Creating Order Options .....	89
Order Options.....	89
Assigning Option Definitions to Catalog Options .....	89
Elements.....	89
Calls .....	90

Parameters.....	90
Returns .....	91
Chapter 7: Setting Up Communication Configurations for Orders .....	93
General Information for Configuration.....	93
What Must Be Specified .....	93
Provider Policies .....	93
Chapter 8: Receiving and Fulfilling Orders.....	95
Order Fulfillment .....	95
Order Identifier .....	95
Line Items .....	95
User Information.....	96
Shipping Address.....	96
Billing Address .....	97
Contact Address.....	97
Name .....	98
Address .....	98
Client Identity .....	98
Appendix A: Acronyms Used in ECHO.....	99
Appendix B: ECHO Path URIs.....	101
Format.....	101
Behavior.....	101
Appendix C: ECHO Error Handling.....	103
Appendix D: Ingest DTDs .....	107
Collection Metadata DTD.....	107
Granule Metadata.....	114
Update Metadata DTD .....	119
Browse Metadata DTD .....	120
Appendix E: Best Practices for Preparing Your Metadata .....	121
Tips .....	121
Submission of Large Numbers of Files and Granules .....	121
Tools to Help Prepare XML .....	121
Additional Tips .....	121

Factors That Affect Ingest Rates.....	122
Common Errors.....	123
Index .....	125

**This page is intentionally left blank.**

## Table of Code Listings

Code Listing 1: Expression of Temporal Information (Range-Day Time).....	19
Code Listing 2: Full Platform/Instrument/Sensor Description .....	20
Code Listing 3: No Platform Associated .....	21
Code Listing 4: No Instrument Associated.....	22
Code Listing 5: Sensor Only.....	22
Code Listing 6: Collection-Level Additional Attributes .....	23
Code Listing 7: Granule/Collection Association .....	25
Code Listing 8: Granule Information.....	27
Code Listing 9: Sources and Sensors.....	28
Code Listing 10: No Platform Association.....	28
Code Listing 11: No instrument association.....	29
Code Listing 12: Sensor Only.....	29
Code Listing 13: Granule level additional attributes .....	30
Code Listing 14: Measured Parameters .....	31
Code Listing 15: Orbital Information .....	31
Code Listing 16: Restriction Flag Set for a Granule.....	36
Code Listing 17: Setting Two-Dimensional Coordinate System Coordinates .....	37
Code Listing 18: Single Point Example.....	39
Code Listing 19: Multiple Points.....	39
Code Listing 20: Single Line Example.....	40
Code Listing 21: Multiple Line Example .....	41
Code Listing 22: Interpreted in Cartesian and Geodetic Systems .....	41
Code Listing 23: Adding Density .....	43
Code Listing 24: Single Polygon .....	44
Code Listing 25: Single Polygon with a Hole .....	46
Code Listing 26: Bounding Box .....	47
Code Listing 27: Polygon with Points in Counter-Clockwise Order.....	49
Code Listing 28: Twisted Polygon .....	50
Code Listing 29: Hole Crosses over the Outer Ring.....	51
Code Listing 30: Polygon Crosses International Dateline .....	53

Code Listing 31: Overlapping Polygons.....	54
Code Listing 32: Inappropriate Data.....	56
Code Listing 33: Incorrect Density.....	57
Code Listing 34: Correct Density .....	58
Code Listing 35: Polygon Covering More Than Half of the Earth.....	59
Code Listing 36: Metadata Update DTD .....	62
Code Listing 41: Sample Browse XML File .....	74
Code Listing 42: Update Browse Example: testbrowseu.xml .....	75
Code Listing 43: Delete Browse Example: testbrowsed.xml .....	76
Code Listing 44: Example of the Input File Error Report .....	77
Code Listing 45: XML Validation Error .....	78
Code Listing 46: Example of the Ingest Summary Report for a Successful Ingest.....	79
Code Listing 47: Error Returned When Input Data Is Older Than the Information Recorded in the ECHO Database.....	80
Code Listing 48: Error Returned Because of Duplicate Identifiers in a Single Round of Ingest.....	81
Code Listing 49: Error Returned When the Spatial Coverage Area Data for a Collection or Granule Is Invalid .....	81
Code Listing 50: Error Returned When Any Piece of Data Associated with a Collection or Granule Has Violated a Data Constraint .....	82
Code Listing 51: Error Returned When a Collection or Granule Identifier Sent for Deletion Does Not Exist in the ECHO Database .....	82
Code Listing 52: Error Returned Because Collection Referenced by Input Granule(s) Does Not Exist in the ECHO Database.....	83
Code Listing 53: Non-XML File Format.....	83
Code Listing 54: Non-MetadataUpdate XML File.....	84
Code Listing 55: Invalid Metadata Update.....	84
Code Listing 56: Catching Exceptions from ECHO.....	104

# Preface

This document describes the Version 9.0 operational release of the Earth Observing System (EOS) Clearinghouse (ECHO) system.

## Conventions

All references to time are in Universal Time Coordinated (UTC).

Data Partners are also referred to as Data Providers.

Client Partners are also referred to as Client Developers.

Words in **bold** text are key words or concepts.

---

Programming examples use a fixed width font and this color font.

---

---

Comments (denoted by // within examples) use this color font.

---

*Best practices or warnings appear in italicized, boxed text.*

**This page is intentionally left blank.**

# Chapter 1: Before You Begin

The primary reason for designing the EOS ClearingHOuse (ECHO) was to increase access to Earth science data and services by providing a system with a machine-to-machine interface, that is, an Application Programming Interface (API).

ECHO functions as a metadata clearinghouse of Earth science metadata for a wide variety of partners, enabling the science community to exchange information. Data Partners provide the ECHO community with metadata representing their Earth science data holdings. ECHO technology in turn provides services for Client Partners and Data Partners and supports efficient discovery and access to Earth science data.

ECHO also functions as an order broker for the data, and offers services applied to that data. ECHO provides a portal on the internet where ECHO clients can search the metadata for information they wish to order.

Client applications can access data holdings via order distribution or online access. Data Partners retain complete control over what metadata are represented in ECHO including inserting new metadata, modifying existing metadata and removing old metadata, and controlling access to their metadata.

## Tasks That You Will Perform as a Data Partner

Usually performed in the order shown below:

- Providing metadata for ingest into the ECHO database
  - Validating your ingested metadata
  - Controlling access to your metadata through:
    - Access Control Rules
    - Setting visibility restrictions
  - Creating Order Options
    - Creating order options definitions
    - Assigning option definitions to catalog items
  - Setting up your (Data Provider) communication configurations for orders
    - Specifying URL and Port for order communications
    - Specifying communication retry attempts (in case of communication failure) and specifying time interval between retries
  - Receiving and Fulfilling Orders
- 
- Chapter 3
- Chapter 4
- Chapter 5
- Chapter 6
- Chapter 7
- Chapter 8

## Skills You Will Need as a Data Partner

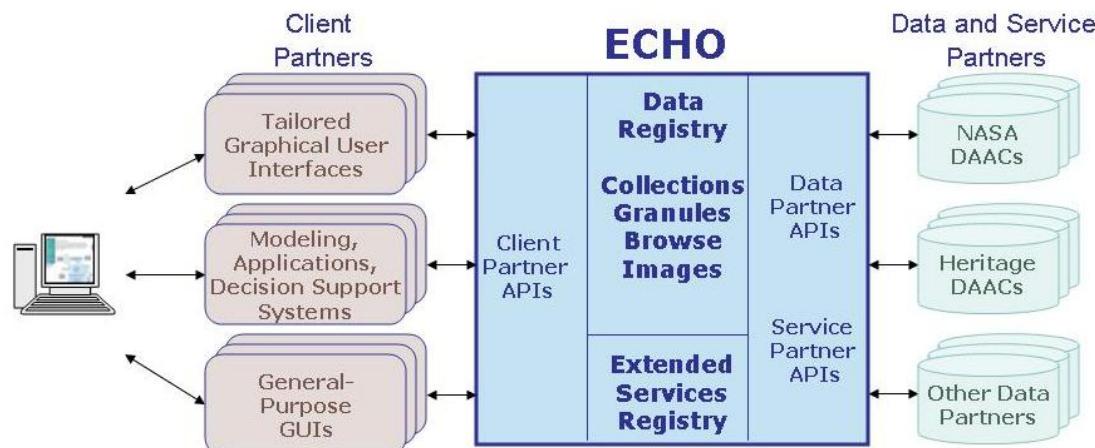
Since ECHO uses platform-independent web service definitions for its API, there are no requirements for a client programming language. All examples in this document are in snippets of Java code; however, the code samples provided could be translated to any web service capable language.

As an ECHO Data Partner, you need to be familiar with basic software development and Service Oriented Architecture (SOA) concepts such as:

- XML and XML Schema (XSD)
- Web Service Definition Language (WSDL)
- Service-based Application Programmer's Interface (API)

## ECHO Concept and Design

Internally, ECHO specifies APIs and provides middleware components, including data and service search and access functions, in a layered architecture. The figure below depicts the ECHO system context in relation to its public APIs.



ECHO allows Data Partners to cache copies of their metadata within it. Data Partners have complete control over what metadata ECHO represents on their behalf. You, as a Data Partner, can insert new data, modify existing data and remove old data.

All ECHO metadata is stored in an Oracle database with spatial extensions. The metadata model is derived primarily from that used by the Earth Observing System Data and Information System (EOSDIS) Core System (ECS). For more details about the ECHO model, refer to [http://www.echo.nasa.gov/data\\_partners/ECHO1\\_70.shtml](http://www.echo.nasa.gov/data_partners/ECHO1_70.shtml).

Key features of the ECHO architecture are:

- *Ease of Partner Participation* – Designed to be low-cost and minimally intrusive, ECHO offers a set of standard ways for partners to interface with the system and a metadata exchange approach that accommodates existing partners and technology.

- *Data Model Consistency* – To mitigate the risk of being unable to match all possible partner data models, ECHO has prototyped a Metadata Mapping Tool to translate non-standard formats upon ingest into ECHO.
- *Open System/Published APIs* – To accommodate independent ECHO clients, ECHO uses an open system approach and publishes domain APIs. These APIs are independent of the underlying transport protocols used. ECHO communicates using WS-I Basic Profile v1.0 compliant web services. This API is located at <http://www.echo.nasa.gov/reference/reference.shtml>.  
Interactions with ECHO may involve user interactions in real time or may be machine to machine.
- *Evolutionary Development* – The ECHO system is being developed incrementally to allow for insight and feedback during the development cycle. Industry trends are followed and the use of commercial, off-the-shelf (COTS) products is optimized.

### ECHO as a Spatially Enabled Metadata Search and Order System

Oracle enables the ECHO system to interact with spatially enabled Earth science metadata by use of spatial extensions into the system and business logic within the system that understands how to interact with that metadata. In addition, a second ECHO interface (auto-ingest) allows metadata updates to go directly into the database, bypassing the message-passing API. The File Transfer Protocol (FTP) server is configured to receive these update files, which are expressed in XML conforming to two DTDs, one for granules (or inventory) and one for collections (or datasets).

- The FTP server is located here:  
<FTP://ingest.echo.nasa.gov>
- All DTDs are defined here:  
<http://www.echo.nasa.gov/reference/reference.shtml>

Oracle's spatial capabilities support queries for ECHO metadata whose spatial extent is described within the system. A Data Partner can define the spatial extent of a granule or a collection with different spatial constructs (for example, point and polygon). A Client Partner can then construct a search using a point, a line, or a polygon (or multiple polygon) spatial type, and ECHO responds with data whose spatial region intersects the described region.

ECHO provides services for interacting with its **Catalog** of metadata. Queries can be performed in a number of ways; result formats can be specified, and the resulting data sets can be incrementally accessed so that large return sets can be handled gracefully. ECHO also supports constructing, submitting, and tracking orders for the data that the metadata represents. ECHO supports both an embedding of a Uniform Resource Locator (URL) within the metadata for accessing the data (which the client simply accesses via Hypertext Transfer Protocol [HTTP]), and a more complicated order process in which quotes and order options are accommodated.

ECHO incorporates the ECS concept of granules and collections and defines separate DTDs for updating each, under the assumption that granules will indicate which collection is considered their “primary” collection. “Primary collection” means the collection that owns the granule.

A **collection** is a grouping of granules that all come from the same source, such as a modeling group or institution. Collections have information that is common across all the granules they contain and a template for describing additional attributes not already part of the metadata model.

A **granule** is the smallest aggregation of data that can be independently managed (described, inventoried, and retrieved). Granules have their own metadata model and support values associated with the additional attributes defined by the owning collection.

A third type of metadata, which is not spatially enabled but useful, is **browse metadata**, which provide a high-level view of granule or collection metadata and cross-referencing to other granules or collections.

## Security

The ECHO system supports Secure Sockets Layer (SSL)-based communication, which a client can use to pass passwords or other sensitive information securely. Internally, the systems are fire-walled to prevent unintended access.

## Supported Platforms

The ECHO system supports clients capable of initiating an HTTP connection from a variety of programming languages.

## ECHO Capability and Functionality

ECHO provides an infrastructure that allows various communities to share tools, services and metadata. As a metadata clearinghouse, it supports many data access paradigms such as navigation and discovery. As an order broker, ECHO forwards orders for data discovered through the metadata query process to the appropriate Data Partners for order fulfillment. As a service broker, ECHO decentralizes end user functionality and supports interoperability of distributed functions.

Although this Guide focuses on the needs of Data Partners, ECHO supports the following different, nonexclusive types of Partners:

- *Data Partners* – Organizations that supply metadata representing their data holdings to the ECHO database.
- *Client Partners* – Organizations that participate by developing software applications to access the Earth science metadata in the ECHO database.
- *Service Partners* – Organizations that participate by advertising their Earth science-related services to the user community via ECHO, which maintains service descriptions in a Service Catalog (either special services, or services that are available as an option on a selected set of granules/collections) and support the user in ordering those services.

- *Extended Service Partners* – Organizations that participate by providing a central location for registration, classification, and maintenance of Earth science services, interfaces, GUIs, and advertisements.

ECHO addresses science user needs through a set of well-defined and open interfaces upon which the user community can build its own client applications. In this way, ECHO supports extendable, flexible user interfaces, allowing industry and the science community to drive the progress of available Earth science applications. For more complete information about client applications, refer to the companion piece to this Guide, the *ECHO 9.0 Client Partner's Guide*.

The ECHO approach allows users to build their own user interfaces to ECHO, rather than being limited to the data search and order system provided by NASA. For Data Partners, ECHO offloads the burden of providing the system resources required for searching and gives users the flexibility to support community-specific services and functionality. ECHO's interoperability features allow all participants to benefit from the distributed development of functions, again reducing dependence on NASA resources.

### Benefits to Data Partners

ECHO'S open system provides Earth science data and services to a large, diverse pool of users, enabling scientific community interaction and collaboration. ECHO benefits Data Partners in the following ways.

- Control in the hands of the data partner
- Automate mapping between your metadata and ECHO catalog metadata
- Makes data resources available to a wide ranges of potential users
- Virtual “co-location” with other data sources and services
- Common data language
- Enable loosely coupled application solutions

**This page is intentionally left blank.**

## Chapter 2: The Basics

This chapter describes the basic terms and concepts used in subsequent discussions of the Web Services API and the ingest process.

### Locating the ECHO Web Services

To access a particular service through the ECHO Web Services API, refer to the Reference page of the ECHO website (<http://www.echo.nasa.gov/reference/>).

The table below shows each ECHO service and a brief description of its capabilities. The ECHO 9.0 Web Services API documentation describes in detail the services along with their operations and parameters, and is currently available online at:  
<http://api.echo.nasa.gov/echo/ws/v9/index.html>.

To access the Web Service Description Language (WSDL) document that describes a service, attach the suffix .wsdl from the API page following this format:  
`http://api.echo.nasa.gov/echo-wsdl/v9/<Service Endpoint>.wsdl`.

**Table 1: Description of Data Partner-Related ECHO Web Services**

Service Name	Description	Service Endpoint
Catalog	Data warehouse searching and exploration	/CatalogService
Data Management	Data Partner service to support ECHO cataloged data	/DataManagementService
Group Management	Data Partner service to organize users into groups for data access control and notification	/GroupManagementService
Order Processing	Data Partner-oriented service to fill and apply a status to user orders	/OrderProcessingService
Provider	Data Partner account creation and maintenance	/ServiceProvider

Service Name	Description	Service Endpoint
Taxonomy	Management interface for data and service classification schemes used by ECHO, Data Partner and Client Partners	/TaxonomyService
User	User account creation and maintenance	/UserService

## ECHO Globally Unique Identifiers (GUIDS)

An ECHO Globally Unique Identifier (GUID) is a mostly random number with a large number of unique keys. A GUID is normally a 16-byte (128-bit) number in hexadecimal form.

ECHO uses GUIDs to identify items such as users, providers, contacts, orders, etc. Client applications use GUIDs to find and operate on items using the ECHO API.

In almost all cases (with the notable exception of the Taxonomy API), the GUID on an item should be null when the item is sent to ECHO to be created. Once ECHO creates the item, it will generate a new GUID for the item and return it to the client.

## ECHO Entities

This section describes several high-level concepts that help you understand the ECHO system. The following is a selected list of entities. For the complete list of ECHO entities, refer to [http://www.echo.nasa.gov/data\\_partners/ECHO3.shtml](http://www.echo.nasa.gov/data_partners/ECHO3.shtml).

Data Partners use Provider User Management Program (PUMP) or the API itself to manage these entities—refer to Chapter 5: Controlling Access to Your Metadata,” beginning on Page 87.

### Users

The most basic entity in the ECHO system is a **user**. Each user is identified by a unique user name. There are two types of users: **registered users** and **guests**. Registered users can save information they plan to use in their next session. Guests have the ability to do many of the things registered users can do, but they cannot count on persistent access to information across sessions in addition to other limitations.

### Roles

ECHO regulates access privileges based on the concept of user roles. User roles are a way to grant a user access to the system. These roles facilitate greater flexibility with operation-level authorization and allow certain users the ability to have more than one role without having more than one account in the system.

As a Data Partner, your role is the **provider role**. You may have one or more provider roles, each of which is associated with one provider in the ECHO system. Your provider roles allows you to access and update information about the providers with which they are associated. For example, if you have a provider role for Oak Ridge National Laboratory (ORNL), then you can use the UpdateContact operation to update the contact information for ORNL.

You must use the provider context to tell the system which role you want to represent.

*Note: If you are associated with only one provider role, the system assumes the provider associated with that one provider role is specified in the user's "provider context."*

## Queries (Used Primarily by Client Partners)

Client Partners use queries in the ECHO system for search and retrieval of science metadata stored by ECHO. For a detailed discussion of queries, refer to Chapter 4, “Querying for Earth Science Metadata,” in the *Client Partner’s User’s Guide*.

## Catalog Items

A **catalog item** is any metadata item (granule or collection) that is available for ordering from the ECHO system. The results of a query may return several granules or collections. Catalog items are identified by a **catalog item GUID** (CatalogItemId, which is an assigned XML metadata tag).

To see the possible required/optional options for a catalog item, invoke the **GetCatalogOrderInformation** operation on the **OrderManagementService** passing the catalog item GUIDs of interest.

You (Data Providers) can establish your order policy and provide ECHO with a list of your granules/collections that are orderable or searchable. Another option is for a provider to follow ECHO’s rule to provide orderable notification by giving the price for each orderable item even it is \$0.00. Note that if a URL is provided, it is assumed that the client can simply retrieve the data from that URL as a direct link or obtain the data accessing instruction via the URL. For more information about creating your order policy, refer to “Chapter 6: Creating Order Options.”

## Orders

An **order** is a collection of **catalog** items that a client wants to have and will order from a Data Provider. Each item in the order is associated with a quantity and any options available to that item. Within ECHO, a user creates an order and then adds, deletes, and updates each item in the order before submitting the order to ECHO. Orders can also be created and submitted in a single web service API call (CreateAndSubmitOrder).

The **collection** of catalog items that comprises an order does not have to belong to one provider, but can span many providers. When organizing providers and catalog items within an order, another concept called a “provider order” is used. An order can consist of one or more provider orders. Each provider order can consist of one or more catalog

items that belong to the same provider. To identify a specific provider order, you need the GUID of the order that includes that provider order and the GUID of the Data Partner associated with that provider order.

When a full order is submitted, ECHO splits the user's order into separate provider orders and submits each provider order to the associated Data Provider.

The **OrderManagementService** allows users to create and change orders, provider orders, or individual catalog items. Once the **SubmitOrder** operation is executed for a certain order within the **OrderManagementService**, the user can no longer execute any further changes on that order. However, a registered user can look at the current and historical status of any of their submitted orders.

Once a provider order is submitted to the appropriate provider, the status of that order can be changed in two ways:

- The Data Provider can send an immediate response, whether they will or will not accept the order, to an order submission.
- The Data Provider can wait and asynchronously use the **OrderProcessingService** to change the status of an order after they have had time to process the order.

Most Data Providers use Provider User Management Program (PUMP) for group management, user management, provider information, provider contacts, provider policies, provider orders, data management, user lookup, and storing your address, phone numbers, and other options. : For more information on the Order Management Service, refer to the “Managing Data Access Rules” section of “Chapter 5: Controlling Access to Your Metadata,” Page 87.

## Order Options

Data Partners generally use Provider User Management Program (PUMP) or the API itself to set up order options. To download PUMP and its user's guide, go to [http://www.echo.nasa.gov/data\\_partners/data\\_tools.shtml](http://www.echo.nasa.gov/data_partners/data_tools.shtml).

Use **order options** to describe the structure of the data to request of the client as well as how to display the order form to the client. **Option definitions** contain a name, scope, and deprecated flag, and an ECHO Form. Every option definition for a provider has a unique name. The **scope** indicates whether it is a system-level option definition or provider-level option definition. The **deprecated flag** indicates that you, as a Data Partner, have made that option definition obsolete. The deprecated flag indicates to clients that they should no longer use the option definition. The form part of the option definition contains an ECHO Form.

The **OptionSelection** is the data from the client for a specific option definition. The selection should be put in the content part of the option selection. Options are also used in ECHO to describe authenticators and options for ordering data from a provider.

For more information on Order Options, refer to “Chapter 6: Creating Order Options,” Page 89.

## Authenticators

While most items that can be ordered through ECHO’s data broker functionality are available to the entire user community, some items are restricted. Different Data Partners use different mechanisms to determine if the person ordering restricted data is allowed to access that data. Some providers use a password, others an authentication key (which is a string). To accommodate these differing needs, ECHO provides the authenticator mechanism. Data Partners are allowed to define authenticator structure using options. Clients can present the authenticator in a template form to the user. The client can then create a named list of authenticators on behalf of the user for each provider using operations in the user account service. ECHO allows for a registered user to have more than one authenticator stored for a provider since it is possible that different projects that the user represents may have different authenticators. There are operations for managing this list. Once the list is created, any client can set which one (by name of the authenticator) should be used for a given order. If none is set, then an authenticator is not used. ECHO will send the authenticator along with the order. An order adapter can be used to convert the authenticator from ECHO’s format into the information needed for accessing a provider’s existing interface if needed.

## Groups

The term **groups** refers to an aggregating mechanism in ECHO that allows Data Partners to associate a Group name with a given set of users. When a group is created, the group’s owner specifies an ECHO user to be the group’s manager. However, group managers can be added and removed after creation by other group managers. After becoming a member of a group, a user can be granted access to restricted metadata via the Data Management Service—refer to Chapter 5: Controlling Access to Your Metadata”on Page 85 for details.

## Conditions

Conditions represent a partial equation to be evaluated as part of the Access Control (ACL) honoring system. The type of the condition defines the evaluation process. Temporal Conditions use a date range, so that a date associated with a granule can be compared against the date range to check for applicability of the condition. The primary use of conditions is to facilitate reuse among data rules. The same temporal condition can be used by both a restriction and a permission to control access to metadata. To extend a time range, you only have to change one **TemporalCondition** as opposed to changing multiple data rules. Another type of condition is **RestrictionFlag**, which can be used to restrict access to collections or granules based on the value of a **RestrictionFlag** metadata field. For example, this might be used to control access based on a granule’s science quality.

## Rules

Rules include conditions and provide a complete evaluation. Rules define which specific data is to be controlled, as well as the condition to use for evaluating whether the data should be controlled. Rules also contain a comparator, which is a key part of rule evaluation. Lastly, rules contain data including **ActionType** (describes which actions the

rule applies to), and in the case of a permission, a **GroupName** (describes which Group the permission applies to). **Restrictions** (one type of a rule) apply to the global ECHO population, and **permissions** (the other type of a rule) apply to a specific group. Refer to Chapter 5: Controlling Access to Your Metadata” on Page 85 for details about using rules for data management.

### Extended Services

ECHO Extended Services allows partners to advertise and register web service interfaces, implementations, GUIs, and advertisements.

For more information about Extended Services and Service Partners, refer to the forthcoming *ECHO Service Partner’s Guide*—check the ECHO website for availability.

# Chapter 3: Ingest

## Ingest Overview

Metadata ingest refers to the process of inserting, deleting, or updating metadata in the ECHO database and affects only that Data Partner's specific metadata. The main tasks for the metadata ingest process are loading bulk metadata and updating the ECHO database. Although there are slight variations from one Data Partner to another in the transmission process and interaction with ECHO Operations, the process includes the following:

1. Use FTP Push to send the metadata to a folder at `ingest.echo.nasa.gov`, the input holding area for operational ingest (`ingest.echo.nasa.gov`). There is also a Partner Test System FTP site (`ingest-test.echo.nasa.gov`).
2. ECHO checks input folders for new or updated metadata every five minutes. When it detects your metadata, various scripts will run automatically to prepare it for updating the ECHO database.
3. The ingest process will add your metadata to the ECHO database immediately unless there are metadata files in the queue, in which case your metadata will be placed in a “staging area” until ECHO has finished ingesting the existing queue.
4. When ingest is complete, you will receive notification, usually via e-mail; if there were errors in the metadata, you will receive an error report notification.

The metadata input includes Collection, Granule, Update, and Browse metadata. It must be in XML format and conform to the ECHO DTDs, which you can locate on the <http://www.echo.nasa.gov/reference/reference.shtml> page and in “Appendix D: Ingest DTDs,” Page 107.

If you generate the XML file for metadata ingest using your own DTD, then you must map your DTD to ECHO’s DTD and convert your XML file before sending it into ECHO (unless your organization has made other arrangements with ECHO Operations).

Before ingest can begin, ECHO Operations must assign each Data Partner a database account to host metadata. Data Partners receive their own unique provider names, determined by agreement between each Data Partner and the ECHO Operations team.

## Automated Operational Ingest

### Rules for Automated Ingest

Automated ingest of metadata submissions will occur except in the following instances:

1. If you do not provide granule metadata, bulk URL inserts or deletes received for ingest will not be processed. This is to prevent unnecessary errors.
2. If you provide collection metadata, it will be ingested before any other metadata in your metadata submissions. needed before ingest of other metadata can proceed.
3. If you provide an ingest submission with more than 250,000 granules, ingest must be manually managed to prevent your metadata from continuously using an excessive amount of the processing pipeline.
4. Processing of historical metadata is not automatic. It has to be manually managed to allow all Data Partners the use of the ingest pipeline.
5. Your new input metadata will not be ingested automatically until all metadata and ingest issues have been resolved so that ingest can proceed to completion on a regular basis.
6. If ECHO encounters any ingest halts in automatic ingest from your input submission, automatic ingest will be turned off for that input until the issue causing the failure is determined and resolved.

For information about **Best Practices** and methods to avoid errors for automated ingest, refer to

### Notification of Metadata Transmission

No specific notification of metadata transmission is required, but it is helpful to ECHO Operations, which monitors the Data Provider's FTP repository for file transmission and completion. The automated ingest process may also perform monitoring.

## Considerations for Ingest Metadata

### Additional Attributes

The ECHO system stores the additional attributes, also known as product-specific attributes (PSAs), including name and value, exactly the way it receives them. The ECHO system does not do a unification or mapping of the additional attributes' names among the Data Partners.

### Metadata Mapping/Ingest Process

The ECHO ingest process directly accepts and processes metadata input files conforming to the ECHO DTD. Metadata input files conforming to other DTDs require that you perform an XML-to-XML conversion before submitting the files to the ECHO ingest process. Contact the ECHO Operations team for more information.

When the ECHO ingest process detects potential input files in your FTP input directory, it will then examine and validate each file, checking to see if the first line of text contains the <!xml....> declaration.

- **If this line is not present**, the ingest process will reject the input file.
- **If this line is present**, the ingest process will verify that the file is located in the appropriate directory. For example, if you have placed a granule XML file in the collection (rather than the granule) FTP input directory, the ingest process will reject the input file.
- Finally, the ingest process will validate the input file with its corresponding DTD. If this validation fails, the ingest process will reject the input file. Whenever the ingest process rejects an input file, it will record an error and send it to you via your preconfigured provider contact e-mail address. It will send the same information to the ECHO Operations staff as well.

Once an input file has been successfully validated, the ECHO ingest process will load the metadata into the ECHO database and update the operational database tables, making the metadata available for public search. The ingest process will send you an ingest summary e-mail via your preconfigured provider contact e-mail address and will send the same information to the ECHO Operations staff as well.

ECHO applies additional constraints at the time of ingest to maintain granule metadata integrity. ECHO will reject a granule that violates any of the following constraints:

- Collection metadata must be in the ECHO database before ECHO will accept the collection's granule metadata.
- A granule's Additional Attributes must be a subset of its collection's Additional Attributes.
- A granule's platform/instrument/sensor configuration must a subset of its collection's platform/instrument/sensor configuration.
- A granule's instrument operation mode must be a subset of its collection's instrument operation mode.
- A granule's analysis source must be a subset of its collection's analysis source.
- A granule's campaign must be a subset of its collection's campaign.

### **Creating ECHO-Compatible XML Files**

Generate your metadata XML files using ECHO collection and granule DTDs. ECHO has its own DTDs defined for collection and granule metadata, which you can download from <http://www.echo.nasa.gov/reference/reference.shtml>.

## ECHO Collections

The complete ECHO Collection DTD is available from the ECHO website at <http://www.echo.nasa.gov/reference/reference.shtml> or refer to “Appendix D: Ingest DTDs.” The elements listed below are defined at <http://www.echo.nasa.gov/datapartners/ECHO4.shtml>.

### Basic Collection Information

The **required elements** for collection metadata are:

- ShortName
- VersionID
- InsertTime
- LastUpdate
- LongName
- DataSetID
- CollectionDescription.

**Additional elements** include:

- VersionDescription
- RevisionDate
- SuggestedUsage1
- SuggestedUsage2
- ProcessingCenter
- ArchiveCenter
- CitationforExternalPublication
- CollectionState
- MaintenanceandUpdateFrequency
- ProcessingLevelID

The metadata in these elements can help describe a provider's collection metadata for use in searches.

ECHO also requires that you provide a **DataSetID**, which is a unique identifier, to identify a collection. In most cases this is the **ShortName** and **VersionID** identify the collection. **VersionID** refers to your (the Data Partner's) metadata ID.

**InsertTime** is the day and time you inserted the collection metadata into your database, and **LastUpdate** is the day and time you last updated your collection.

## Spatial Data

Spatial metadata refers to the area of the Earth that the data cover. The ECHO system uses the Oracle database to store spatial information for collections and granules. The spatial coverage area for a granule must be within the spatial coverage area of its primary collection. To ensure that spatial searches will return the correct results, you must prepare the spatial metadata according to the detailed guidelines discussed below.

The ECHO system accepts both Cartesian, Geodetic, and Orbit coordinate systems; refer to the table on the following page for supported spatial data types and guidelines for limitations to granularity. You will choose a coordinate system based on the size and projection of the original data of the spatial area covered.

With the Oracle Cartesian coordinate system, the acceptable spatial data types include Point, Circle, Line, Bounding Box and Polygon. With the Oracle Geodetic coordinate system (World Geodetic System 84), the spatial data types supported include Point, Line and Polygon.

**Table 2: Supported Spatial Data Types**

Spatial Data Type	Cartesian	Geodetic	Orbit	Guidelines and Restrictions
Point	✓	✓	✓	
Circle (Not supported as a Search criterion)	✓	✗	✗	Oracle spatial uses three points on the circumference of the circle.  Any spatial data received as a circle expressed in center-latitude, center-longitude, and radius will not be stored as Oracle spatial data. Instead, it will be stored in a regular relational table.
Line	✓	✓	✓	A line that has vertices across the International Date Line or across the poles is invalid spatial data for the flat Cartesian coordinate system. These restrictions do not apply to Geodetic and Orbit coordinate systems.
Bounding Box	✓	✗	✗	Since neither the Geodetic nor the Orbit coordinating systems support the bounding box type, the ECHO system stores bounding box data as a polygon with four vertices (in the flat Cartesian coordinate system).

Spatial Data Type	Cartesian	Geodetic	Orbit	Guidelines and Restrictions
Polygon	✓	✓	✓	<p>A polygon's vertices must be stored in order of vertex connection. Provide the vertices in clockwise order. No consecutive vertices may have the same latitude and longitude, that is, no repeating points.</p> <p>Keep in mind that a polygon that has vertices across the International Date Line or across the poles is invalid spatial data for both the flat Cartesian and the Geodetic coordinate systems.</p> <p>No polygon should cover more than half the Earth or span half the Earth in both the flat Cartesian and the Geodetic coordinate systems.</p>

The Oracle Geodetic model uses the great circle distance to connect two vertices to construct a polygon area or line. If there is not enough density (that is, the number of points) for a set of vertices, then the line or the polygon area might be misinterpreted or the metadata might be considered invalid.

The ECHO system will not manipulate any of the spatial input metadata. You are responsible for the correctness and integrity of your spatial metadata. Prior to sending your metadata to ECHO for ingest, notify ECHO Operations as to which coordinate system (Cartesian or Geodetic) you used.

To prepare your metadata such that ECHO can support polar search and Geodetic search, observe the following:

- The spatial metadata must be stored in one of three possible types: point, line, or polygon. Choose an appropriate metadata type for your spatial metadata.
- Use only one coordinate system per data set.
- Provide spatial data with appropriate density if using the Geodetic model.
- Ingest for a metadata record will fail if any spatial metadata input is invalid under Oracle Spatial—refer back to Table 2: Supported Spatial Data Types on Page 17 if needed.
- By default, ECHO assumes the maximum spatial coverage area for any Data Partner is the whole Earth (−180 to 180 for longitude and −90 to 90 for latitude). Specify the maximum spatial coverage area only if it varies from the default.
- Keep in mind the restrictions regarding lines and polygons described in Table 2: Supported Spatial Data Types on Page 17.
- Provide the resolution (the default is 0.0001) for both latitude and longitude for your spatial metadata in advance. Your metadata generation application must define the resolution for vertices' degree representation. If any two vertices'

difference is less than the resolution, those two vertices will be considered identical, which might cause the spatial metadata to be invalid for Oracle Spatial.

*Note: At this time, ECHO does not allow a spatial search on a granule or collection that uses a circle spatial coverage.*

## Temporal Data

Temporal metadata refer to a date and time associated with a collection and are essential search criteria for collections.

The ECHO system does not require any specific date format but does require that you use one and only one of the conventional date formats for all data that reflect date/time information. You must notify the ECHO Operations team, in advance, which date format you decide to use. The value you specify in the temporal information specific to acquisition, such as a collection's range (the beginning date and range ending date), must conform to GMT/UTC.

Your collection may be associated with one or more of the following three types of temporal expressions: SingleDateTime, RangeDateTime and PeriodicDateTime. You must use GMT/UTC for your temporal information.

**Code Listing 1: Expression of Temporal Information (Range-Day Time)**

```
<Temporal>
  <TimeType>UTC </TimeType>
  <DateType>Gregorian </DateType>
  <TemporalRangeType>Continuous Range</TemporalRangeType>
  <PrecisionofSeconds>1</PrecisionofSeconds>
  <EndsatPresentFlag>Y</EndsatPresentFlag>
  <RangeDateTime>
    <RangeBeginningDate>1998-01-01 00:00:00.0</RangeBeginningDate>
    <RangeBeginningTime>00:00:00.000000</RangeBeginningTime>
    <RangeEndingDate>1998-01-01 00:00:00.0</RangeEndingDate>
    <RangeEndingTime>00:00:00.000000</RangeEndingTime>
  </RangeDateTime>
</Temporal>
```

The format of the date expression in this example is: yyyy-mm-dd hh24:mi:ss.ms(1). This format allows four digits for year, two digits for month, two digits for day, two digits for hour in a 24-hour system, two digits for minute, and two digits for second with one digit for precision of the second. You may express your date information in a different format and ECHO will process the information accordingly. However, once you define the format and communicate it to the ECHO Operations team, all your future date information must follow that format. Otherwise, ECHO will not be able to process the information, resulting in a NULL entry.

## Sources and Sensors

ECHO adopts a layered representation of source and sensor information for the collection. A source and sensor layer is defined as:

---

```
Platform->* Instrument->* Sensor
```

---

A collection could be associated with zero (0) or more platforms; each platform could contain zero or more instruments, and each instrument could contain zero or more sensors. There might be characteristic parameters associated with the platforms, instruments, and/or sensors for the collection. There might also be an operational mode associated with instruments. When you do not have the platform concept within your collection, the input for platform short name should be “No Platform Associated.” When you do not have the instrument concept for your collection, the input for instrument short name should be “No Instrument Associated.”

### Code Listing 2: Full Platform/Instrument/Sensor Description

---

```
<Platform>
  <PlatformShortName>Terra</PlatformShortName>
  <Instrument>
    <InstrumentShortName>MODIS</InstrumentShortName>
    <Sensor>
      <SensorShortName>MODIS</SensorShortName>

      <SensorLongName>Cross-track Scanning Radiometer</SensorLongName>
      <SensorTechnique>Radiometry</SensorTechnique>
    </Sensor>
    <OperationMode>Operation Mode</OperationMode>
  </Instrument>
</Platform>
<Platform>
  <PlatformShortName>Terra</PlatformShortName>
  <PlatformLongName>First EOS Polar Orbiting Satellite, 10:30 AM
Descending Equator Crossing</PlatformLongName>
  <PlatformType>Spacecraft</PlatformType>
  <PlatformCharacteristic>
    <PlatformCharacteristicName>
      EquatorCrossingTime
    </PlatformCharacteristicName>
    <PlatformCharacteristicDescription>Local time of the equator
crossing and direction (ascending or
descending)</PlatformCharacteristicDescription>
    <PlatformCharacteristicDataType>varchar
  </PlatformCharacteristicDataType>
    <PlatformCharacteristicUnit>Local Mean
Time</PlatformCharacteristicUnit>
    <PlatformCharacteristicValue>10:30,
descending</PlatformCharacteristicValue>
  </PlatformCharacteristic>
<Instrument>
  <InstrumentShortName>InstrumentName</InstrumentShortName>
  <InstrumentLongName>
```

---

---

```
Moderate-Resolution Imaging Spectroradiometer
</InstrumentLongName>
<InstrumentTechnique>Imaging
Spectroradiometry</InstrumentTechnique>
<Sensor>
<SensorShortName>InstrumentName</SensorShortName>
<SensorLongName>Cross-track Scanning
Radiometer</SensorLongName>
<SensorTechnique>Radiometry</SensorTechnique>
</Sensor>
</Instrument>
</Platform>
```

---

#### Code Listing 3: No Platform Associated

---

```
<Platform>
<PlatformShortName>No Platform Associated</PlatformShortName>
<Instrument>
<InstrumentShortName>Acronym for Instrument</InstrumentShortName>
<InstrumentLongName>Full name of Instrument</InstrumentLongName>
<InstrumentTechnique>Imaging
Spectroradiometry</InstrumentTechnique>
<Sensor>
<SensorShortName>Acronym for Instrument</SensorShortName>
<SensorLongName>Cross-track Scanning Radiometer</SensorLongName>
<SensorTechnique>Radiometry</SensorTechnique>
</Sensor>
</Instrument>
</Platform>
```

---

---

**Code Listing 4: No Instrument Associated**

---

```
<Platform>
  <PlatformShortName>Spacecraft being used</PlatformShortName>
  <PlatformLongName>First EOS Polar Orbiting Satellite, 10:30 AM
  Descending Equator Crossing</PlatformLongName>
  <PlatformType>Spacecraft</PlatformType>
  <PlatformCharacteristic>

    <PlatformCharacteristicName>EquatorCrossingTime</PlatformCharacteristicName>
      <PlatformCharacteristicDescription>
        Local time of the equator crossing and direction (ascending
        or
        descending)
      </PlatformCharacteristicDescription>

    <PlatformCharacteristicDataType>varchar</PlatformCharacteristicDataType>
      <PlatformCharacteristicUnit>Local Mean
      Time</PlatformCharacteristicUnit>
      <PlatformCharacteristicValue>10:30,
      descending</PlatformCharacteristicValue>
    </PlatformCharacteristic>
    <Instrument>
      <InstrumentShortName>No Instrument
      Associated</InstrumentShortName>
      <Sensor>
        <SensorShortName>MODIS</SensorShortName>
        <SensorLongName>Cross-track Scanning
        Radiometer</SensorLongName>
        <SensorTechnique>Radiometry</SensorTechnique>
      </Sensor>
    </Instrument>
  </Platform>
```

---

**Code Listing 5: Sensor Only**

---

```
<Platform>
  <PlatformShortName>No Platform Associated</PlatformShortName>
  <Instrument>
    <InstrumentShortName>No Instrument
    Associated</InstrumentShortName>
    <Sensor>
      <SensorShortName>Acronym for Sensor</SensorShortName>
      <SensorLongName>Cross-track Scanning
      Radiometer</SensorLongName>
      <SensorTechnique>Radiometry</SensorTechnique>
    </Sensor>
  </Instrument>
</Platform>
```

---

## Keywords

ECHO supports three kinds of keyword associations for collections: discipline keywords, spatial keywords, and temporal keywords. Discipline keywords and spatial keywords must comply with the Global Change Master Directory (GCMD) keywords standard located at <http://gcmd.gsfc.nasa.gov/Resources/valids/index.html>.

## Additional Attributes

Additional attributes are parameters, also known as Provider-Specific Attributes (PSAs) that further describe the collection and are important search criteria for the collection. ECHO supports collection-level additional attributes.

**Code Listing 6: Collection-Level Additional Attributes**

```
<AdditionalAttributes>

<AdditionalAttributeDataType>varchar</AdditionalAttributeDataType>
  <AdditionalAttributeDescription>
    Version of the process software that generated the
    product.
  </AdditionalAttributeDescription>

<AdditionalAttributeName>PROCESSVERSION</AdditionalAttributeName>
</AdditionalAttributes>
<AdditionalAttributes>

<AdditionalAttributeDataType>String</AdditionalAttributeDataType>
  <AdditionalAttributeDescription>Estimated RMS error in
geolocation</AdditionalAttributeDescription>

<AdditionalAttributeName>GEO_EST_RMS_ERROR</AdditionalAttributeName>
</AdditionalAttributes>
<AdditionalAttributes>

<AdditionalAttributeDataType>String</AdditionalAttributeDataType>
  <AdditionalAttributeDescription>Flag set (to 0) if
science_state, the L1A engineering data flag that indicates the
Normal/Test configuration of the MODIS instrument, was set for at
least one scan in the granule.</AdditionalAttributeDescription>
  <AdditionalAttributeName>SCI_STATE</AdditionalAttributeName>
</AdditionalAttributes>
<AdditionalAttributes>

<AdditionalAttributeDataType>String</AdditionalAttributeDataType>
  <AdditionalAttributeDescription>Flag set (to 0) if
science_abnormal, the L1A engineering data ground-set flag that
indicates potentially abnormal science data due to things other
than MODIS (such as maneuvers, data link, etc.), was set for at
least one scan in the granule.</AdditionalAttributeDescription>
  <AdditionalAttributeName>SCI_ABNORM</AdditionalAttributeName>
</AdditionalAttributes>
```

---

```
<AdditionalAttributes>
  <AdditionalAttributeDataType>int</AdditionalAttributeDataType>
    <AdditionalAttributeDescription>The number of the granule for
the day starting at midnight. Example: The granule from 00:00:00
to 00:00:05 will be granule number 1. The granule from 00:01:00
to 00:01:05 will be granule number
13</AdditionalAttributeDescription>

<AdditionalAttributeName>GRANULENUMBER</AdditionalAttributeName>
</AdditionalAttributes>
```

---

## Other Descriptive Information

A collection may also include metadata for the following information:

- Computer Science Data Type (CSDT)
- Contact information
- Campaign information
- Association
- Browse
- Online URL
- Coordinate system and planar information for collection's raw data
- DIF IDs (Data Interchange Format)

For more information about the items listed above, refer to  
<http://www.echo.nasa.gov/datapartners/ECHO4.shtml>.

## Restriction Flag for a Collection

ECHO will restrict collections from being viewed and ordered by anyone except users bearing the provider role by verifying collection's Visibility Flag. By setting the visibility flag to be restricted before ingest of a new collection, ECHO will automatically restrict the new collection from viewing and ordering by all except the user bearing the Data Provider's role. The default restriction flag used for all granules in the collection can be overridden by setting restriction flag on a per granule basis.

Set a value (integer) in **RestrictionFlag** to indicate access constraints on a collection. ECHO references this value, in combination with your own data access rule, to restrict public access to the collection. You may use any range of integers for the RestrictionFlag value and assign your own meanings to these numbers.

- Granules inherit the value of their collection.
- If the collection has no value for the Restriction Flag, then there will be no default value for granules within the collection.

For example, if you decide to use RestrictionFlag for a data quality summary, with a range of values from 0 to 10, with 0 indicating unknown quality, 1 indicating poor quality, and 10 indicating excellent quality, and you have established a data access rule that restricts access to granules with a RestrictionFlag value of less than or equal to 5.

Guest users will only be allowed to view granules with a RestrictionFlag value of 6 or higher. Based on this example, the granule below will not be restricted from any user.

---

```
<RestrictionFlag>7</RestrictionFlag>
<RestrictionComment>default for collection</RestrictionComment>
```

---

For information about setting the Restriction Flag at the granule level, refer to Page 36.

## ECHO Granules

The complete ECHO Granule DTD is available on the ECHO website at <http://www.echo.nasa.gov/reference/reference.shtml>.

### Granule/Collection Association

ECHO's architecture is based on granules. Every granule has to belong to a collection. A collection may contain 0 or more granules. The granule/collection association information is provided via granule metadata input.

**Code Listing 7: Granule/Collection Association**

---

```
<GranuleURMetaData>
  <GranuleUR>SC:MODMGAD.001:81677</GranuleUR>
  <DbID>81677</DbID>
  <InsertTime>2001-09-20 11:43:31.996</InsertTime>
  <LastUpdate>2001-09-20 11:43:31.996</LastUpdate>
  <CollectionMetaData>
    <ShortName>MODMGAD</ShortName>
    <VersionID>1</VersionID>
  </CollectionMetaData>
  ....
</GranuleURMetaData>
```

---

Much information associated with the granule is defined or restricted by its collection. A collection's short name and version ID, combined, uniquely identify the collection to which the granule belongs. A granule can be associated with its collection by using the Data Set ID instead of short name and version number. When a collection is deleted, its granules will be deleted as well.

## Basic Granule Information

The **required elements** for granule metadata are:

- GranuleUR (Granule Universal Reference ID)
- InsertTime
- LastUpdate
- CollectionMetaData (of the associated collection)

**Additional elements** include:

- Data File Size (SizeMBDataGranule)
- Processing Information (ReprocessingPlanned, ReprocessingActual)
- Local Granule Reference (ProducerGranuleID)
- Day Night Flag (DayNightFlag)
- PGE Information (PGEVersionClass)
- Temporal Information (RangeDateTime or SingleDateTime)
- Restriction Settings (RestrictionFlag, RestrictionComment)

The elements listed above are defined at:

<http://www.echo.nasa.gov/datapartners/ECHO4.shtml>

These elements can help describe your granule metadata more completely for the purpose of data searches.

**RestrictionFlag** indicates whether access constraints should be applied when the granule data goes public. If the restriction setting is not explicitly indicated for a granule (via RestrictionFlag), it inherits the restriction setting of its associated collection.

**InsertTime** is the day and time granule metadata was first inserted in your database.

**LastUpdate** is the day and time the granule metadata was last updated in your database.

Your granule may be associated with **SingleDateTime** or **RangeDateTime**, representing the time when granule data were acquired. The granule's temporal information must be in GMT/UTC temporal system, and it must fall within the range of its collection's temporal system definition. Temporal information is an essential piece of metadata for a granule search. Since the temporal system definition is provided by its primary collection, in a granule's metadata, input the temporal system definition if it is not present.

---

**Code Listing 8: Granule Information**

---

```
<GranuleURMetaData>
  <GranuleUR>SC:MODMGGAD.001:81677</GranuleUR>
  <DbID>81677</DbID>
  <InsertTime>2001-09-20 11:43:31.996</InsertTime>
  <LastUpdate>2001-09-20 11:43:31.996</LastUpdate>
  <CollectionMetaData>
    <ShortName>MODMGGAD</ShortName>
    <VersionID>1</VersionID>
  </CollectionMetaData>
  <DataGranule>
    <SizeMBDataGranule>36.332</SizeMBDataGranule>
    <ReprocessingPlanned>further update is
anticipated</ReprocessingPlanned>
    <ReprocessingActual>reprocessed</ReprocessingActual>

    <ProducerGranuleID>MODMGGAD.A2001149.h08v07.003.hdf</ProducerGranuleID>
      <DayNightFlag>Day</DayNightFlag>
      <ProductionDateTime>2001-07-24
12:30:03.0</ProductionDateTime>
      <LocalVersionID>3.0.2</LocalVersionID>
    </DataGranule>
    <PGEVersionClass>
      <PGEVersion>3.0.2</PGEVersion>
    </PGEVersionClass>
    <RangeDateTime>
      <RangeEndingTime>17:20:00.000000</RangeEndingTime>
      <RangeEndingDate>2001-05-29 00:00:00.0</RangeEndingDate>
      <RangeBeginningTime>17:10:00.000000</RangeBeginningTime>
      <RangeBeginningDate>2001-05-29
00:00:00.0</RangeBeginningDate>
    </RangeDateTime>
  </GranuleURMetaData>
```

---

## Sources and Sensors

ECHO adopts a layered representation of source and sensor information for a spatial granule. A source and sensor layers are defined as:

---

```
Platform->* Instrument->* Sensor
```

---

A granule may be associated with zero (0) or more platforms; each platform may contain 0 or more instruments, and each instrument may contain 0 or more sensors. In addition, a granule may define a sensor-level characteristic value for the characteristic parameters defined by its primary collection. In addition to the characteristic parameters, there might be an operational mode associated with instruments independent of its primary collection.

As with collections, when you do not specify the platform within your granule, the input for the platform short name should be “No Platform Associated.” When you do not specify the instrument for your granule, the input for the instrument short name should be “No Instrument Associated.” Although there is the entry for platform/instrument characteristics and a sensor-level characteristics definition in the ECHO granule DTD, following the guidelines stated above for granule sources and sensor information integrity is strongly recommended.

### Code Listing 9: Sources and Sensors

---

```
<Platform>
<PlatformShortName>Terra</PlatformShortName>
<Instrument>
<InstrumentShortName>MODIS</InstrumentShortName>
<OperationMode>Operation Mode</OperationMode>
<Sensor>
<SensorShortName>MODIS</SensorShortName>
<SensorLongName>Cross-track Scanning Radiometer</SensorLongName>
<SensorTechnique>Radiometry</SensorTechnique>
</Sensor>
</Instrument>
</Platform>
```

---

### Code Listing 10: No Platform Association

---

```
<Platform>
<PlatformShortName>No Platform Associated</PlatformShortName>
<Instrument>
<InstrumentShortName>Acronym for Instrument</InstrumentShortName>
<Sensor>
<SensorShortName>Acronym for Sensor</SensorShortName>
<SensorLongName>Cross-track Scanning Radiometer</SensorLongName>
<SensorTechnique>Radiometry</SensorTechnique>
</Sensor>
</Instrument>
</Platform>
```

---

---

**Code Listing 11: No instrument association**

---

```
<Platform>
<PlatformShortName>Acronym for Spacecraft</PlatformShortName>
<Instrument>
<InstrumentShortName>No Instrument
Associated</InstrumentShortName>
<Sensor>
<SensorShortName>Acronym for Instrument</SensorShortName>
<SensorLongName>Cross-track Scanning Radiometer</SensorLongName>
<SensorTechnique>Radiometry</SensorTechnique>
</Sensor>
</Instrument>
</Platform>
```

---

**Code Listing 12: Sensor Only**

---

```
<Platform>
<PlatformShortName>No Platform Associated</PlatformShortName>
<Instrument>
<InstrumentShortName>No Instrument
Associated</InstrumentShortName>
<Sensor>
<SensorShortName>MODIS</SensorShortName>
<SensorLongName>Cross-track Scanning Radiometer</SensorLongName>
<SensorTechnique>Radiometry</SensorTechnique>
</Sensor>
</Instrument>
</Platform>
```

---

The platform, instrument, and sensor referred to by the granule are restricted by its primary collection source/sensor reference and must comply with the GCMD standard, located at <http://gcmd.gsfc.nasa.gov/Resources/valids/index.html>.

## Additional Attributes

Additional Attributes are parameters that further describe the granule and are important search criteria for the granule. The granule's Additional Attributes should fall within the Additional Attributes defined for its associated collection.

**Code Listing 13: Granule level additional attributes**

---

```
<AdditionalAttributes>
  <AdditionalAttribute>

    <AdditionalAttributeName>CalibrationQuality</AdditionalAttributeName>
      <AdditionalAttributeValue>marginal</AdditionalAttributeValue>
    </AdditionalAttribute>
    <AdditionalAttribute>

    <AdditionalAttributeName>MissionPhase</AdditionalAttributeName>
      <AdditionalAttributeValue>A+E</AdditionalAttributeValue>
    </AdditionalAttribute>
    <AdditionalAttribute>

    <AdditionalAttributeName>AveragedBlackBodyTemperature</Additional
    AttributeName>
      <AdditionalAttributeValue>290.0</AdditionalAttributeValue>
    </AdditionalAttribute>
    <AdditionalAttribute>

    <AdditionalAttributeName>NadirPointing</AdditionalAttributeName>
      <AdditionalAttributeValue>Y</AdditionalAttributeValue>
    </AdditionalAttribute>
  </AdditionalAttributes>
```

---

## Measured Parameters

Measured parameters are associated at the granule level only and are important search criteria for the granule. For some providers, the value of certain measured parameters determines the visibility of the granule.

**Code Listing 14: Measured Parameters**

---

```
<GranuleURMetaData>
.....
<MeasuredParameter>
  <MeasuredParameterContainer>
    <ParameterName>MODMGAD</ParameterName>
    <QAStats>
      <QAPercentMissingData>0</QAPercentMissingData>
    </QAStats>
    <QAFlags>
      <AutomaticQualityFlag>Passed</AutomaticQualityFlag>
      <AutomaticQualityFlagExplanation>output file is created
and good</AutomaticQualityFlagExplanation>
      <ScienceQualityFlag>Not Investigated</ScienceQualityFlag>
      <ScienceQualityFlagExplanation>See
http://modland.nascom.nasa.gov/QA\_WWW/release.html for the
Science QA status of this
product.</ScienceQualityFlagExplanation>
    </QAFlags>
  </MeasuredParameterContainer>
</MeasuredParameter>
.....
</GranuleURMetaData>
```

---

## Orbital Information

The following code sample shows how to include information associated with a satellite's orbit:

**Code Listing 15: Orbital Information**

---

```
<GranuleURMetaData>
.....
<OrbitCalculatedSpatialDomain>
  <OrbitCalculatedSpatialDomainContainer>
    <OrbitNumber>7694</OrbitNumber>
    <EquatorCrossingLongitude>-100.187</EquatorCrossingLongitude>
    <EquatorCrossingDate>2001-05-29 00:00:00.0</EquatorCrossingDate>
    <EquatorCrossingTime>17:19:38.910554</EquatorCrossingTime>
  </OrbitCalculatedSpatialDomainContainer>
</OrbitCalculatedSpatialDomain>
.....
</GranuleURMetaData>
```

---

## Other Descriptive Information

A granule may include metadata for the following information:

- Campaign information
- Browse
- Online URL
- Processing/QA/Input historical information

For more information about the items listed above, refer to  
<http://www.echo.nasa.gov/datapartners/ECHO4.shtml>.

A granule can be associated with zero (0) or more browse files, and a browse file can be referenced by more than one granule.

**For more information on browse metadata ingest, refer to the “Examples”**

### **Partial Metadata Insert**

---

```
<?xml version="1.0" encoding="UTF-8"?>
<ProviderAccountService>
    <UpdateMetadata>
        <Granule>
            <Target>
                <ID>SC:MIL2TCAL.002:17872867</ID>
                <ProviderLastUpdateTime>2007-04-12
00:02:46.95</ProviderLastUpdateTime>
                <SaveDateTimeFlag><DONTSAVE/></SaveDateTimeFlag>
            </Target>
            <Add>
                <QualifiedTag>GranuleOnlineResources/OnlineResource/OnlineResourc
eURL</QualifiedTag>
                <MetadataValue>ftp://10dps01u.ecs.nasa.gov/distribution/BRWS/Brow
se.001/2007.04.03/MISBR.A2007093.0137.005.2007101133056.AN.24.hdf
                </MetadataValue>
            </Add>
            <Add>
                <QualifiedTag>GranuleOnlineResources/OnlineResource[OnlineResourc
eURL="ftp://10dps01u.ecs.nasa.gov/distribution/BRWS/Browse.001/20
07.04.03/MISBR.A2007093.0137.005.2007101133056.AN.24.hdf"]/Online
ResourceType</QualifiedTag>
                <MetadataValue>BROWSE</MetadataValue>
            </Add>
            <Add>
                <QualifiedTag>GranuleOnlineResources/OnlineResource[OnlineResourc
eURL="ftp://10dps01u.ecs.nasa.gov/distribution/BRWS/Browse.001/20
07.04.03/MISBR.A2007093.0137.005.2007101133056.AN.24.hdf"]/Online
ResourceMimeType</QualifiedTag>
                <MetadataValue>image/jpeg</MetadataValue>
            </Add>
            <Add>
                <QualifiedTag>OnlineAccessURLs/OnlineAccessURL/URL</QualifiedTag>
```

---

---

```
<MetadataValue>ftp://10dps01u.ecs.nasa.gov/longterm/MISR/MIL2TCAL
.002/2007.04.03/MISR_AM1_TC_ALBEDO_P114_0038776_F04_0008.hdf</Met
adataValue>
    </Add>
    <Add>
<QualifiedTag>OnlineAccessURLs/OnlineAccessURL[URL="ftp://10dps01
u.ecs.nasa.gov/longterm/MISR/MIL2TCAL.002/2007.04.03/MISR_AM1_TC_
ALBEDO_P114_0038776_F04_0008.hdf"]/MimeType</QualifiedTag>
    <MetadataValue>application/x-
hdfeos</MetadataValue>
    </Add>
    <Add>

<QualifiedTag>GranuleOnlineResources/OnlineResource/OnlineResourc
eURL</QualifiedTag>
<MetadataValue>ftp://10dps01u.ecs.nasa.gov/longterm/MISR/MIL2TCAL
.002/2007.04.03/MISR_AM1_TC_ALBEDO_P114_0038776_F04_0008.hdf.xml<
/MetadataValue>
    </Add>
    <Add>
<QualifiedTag>GranuleOnlineResources/OnlineResource[OnlineResourc
eURL="ftp://10dps01u.ecs.nasa.gov/longterm/MISR/MIL2TCAL.002/2007
.04.03/MISR_AM1_TC_ALBEDO_P114_0038776_F04_0008.hdf.xml"]/OnlineR
esourceType</QualifiedTag>
    <MetadataValue>METADATA</MetadataValue>
    </Add>
    <Add>
<QualifiedTag>GranuleOnlineResources/OnlineResource[OnlineResourc
eURL="ftp://10dps01u.ecs.nasa.gov/longterm/MISR/MIL2TCAL.002/2007
.04.03/MISR_AM1_TC_ALBEDO_P114_0038776_F04_0008.hdf.xml"]/OnlineR
esourceMimeType</QualifiedTag>
    <MetadataValue>text/xml</MetadataValue>
    </Add>
    </Granule>
</UpdateMetadata>
</ProviderAccountService>
```

---

## Partial Metadata Delete

```
<?xml version="1.0" encoding="UTF-8"?>
<ProviderAccountService>
    <UpdateMetadata>
        <Granule>
            <Target>
                <ID>SC:MIL3DAE.004:17176039</ID>
                <ProviderLastUpdateTime>2007/04/23
08:32:56</ProviderLastUpdateTime>
                <SaveDateTimeFlag><DONTSAVE/></SaveDateTimeFlag>
            </Target>
            <Target>
                <ID>SC:MIL3DLS.004:17176041</ID>
                <ProviderLastUpdateTime>2007/04/23
08:32:56</ProviderLastUpdateTime>
```

---

```
        <SaveDateTimeFlag><DONTSAVE/></SaveDateTimeFlag>
    </Target>
    <Target>
        <ID>SC:MIL3DAE.004:17176328</ID>
        <ProviderLastUpdateDateTime>2007/04/23
08:32:56</ProviderLastUpdateDateTime>
        <SaveDateTimeFlag><DONTSAVE/></SaveDateTimeFlag>
    </Target>
    <Target>
        <ID>SC:MIL3DLS.004:17176329</ID>
        <ProviderLastUpdateDateTime>2007/04/23
08:32:56</ProviderLastUpdateDateTime>
        <SaveDateTimeFlag><DONTSAVE/></SaveDateTimeFlag>
    </Target>
    <Delete>
        <QualifiedTag>OnlineAccessURLs</QualifiedTag>
<QualifiedTag>GranuleOnlineResources</QualifiedTag>
        </Delete>
    </Granule>
</UpdateMetadata>
</ProviderAccountService>
```

---

### **Partial Granule Update**

---

```
<?xml version="1.0" encoding="UTF-8"?>
<ProviderAccountService>
<UpdateMetadata>
<Granule>
    <Target>
        <ID>SC:MOD14.004:2034404150</ID>
        <ProviderLastUpdateDateTime>2007-04-13
12:07:09.603</ProviderLastUpdateDateTime>
    </Target>
    <Update>
        <QualifiedTag>MeasuredParameter/MeasuredParameterContainer[Parame
terName="MODIS L2 Active Fire
Detection"]/QAFlags/ScienceQualityFlag</QualifiedTag>
            <MetadataValue>Being Investigated</MetadataValue>
    </Update>
</Granule>
<Granule>
    <Target>
        <ID>SC:MOD14.004:2034404150</ID>
        <ProviderLastUpdateDateTime>2007-04-13
12:07:09.603</ProviderLastUpdateDateTime>
    </Target>
    <Update>
        <QualifiedTag>MeasuredParameter/MeasuredParameterContainer[Parame
terName="MODIS L2 Active Fire
Detection"]/QAFlags/ScienceQualityFlagExplanation</QualifiedTag>
            <MetadataValue>Product assessment is
ongoing</MetadataValue>
    </Update>
```

---

```
</Granule>
</UpdateMetadata>
</ProviderAccountService>
```

---

Browse Image Files and Browse Metadata” section on Page 67 of this chapter.

### Restriction Flag for a Granule

Set a value (integer) in **RestrictionFlag** to indicate access constraints on a granule. ECHO references this value, in combination with your own data access rule, to restrict public access to the granule. You may use any range of integers for the RestrictionFlag value and assign your own meanings to these numbers.

- If you indicate a Restriction Flag setting for a granule, it will override the default value for its collection.
- If you do not indicate a Restriction Flag setting for a granule, it will inherit the value of its collection.
- If the collection has no value for the Restriction Flag, then there will be no default value for granules within the collection.

For example, if you decide to use RestrictionFlag for a data quality summary, with a range of values from 0 to 10, with 0 indicating unknown quality, 1 indicating poor quality, and 10 indicating excellent quality, and you have established a data access rule that restricts access to granules with a RestrictionFlag value of less than or equal to 5. Guest users will only be allowed to view granules with a RestrictionFlag value of 6 or higher. Based on this example, the granule below will not be restricted from any user.

#### Code Listing 16: Restriction Flag Set for a Granule

```
<RestrictionFlag>8</RestrictionFlag>
<RestrictionComment>quality summary value of 8
(good)</RestrictionComment>
```

*Note: Restriction flag set for a granule overrides the default Collection value.*

### Spatial Representations, Coordinates and Projections

For a collection or granule, the spatial area coverage is one of the most important and basic search criteria for Earth science data, although it is not required. To submit the metadata so that spatial area coverage can be used as part of a basic search, apply the concepts discussed in this page through Page 49.

ECHO supports multiple spatial representations. Each provider can have different spatial representation; however, each collection must define a single spatial representation for its granules.

ECHO supports three coordinate systems for spatial data. Each Data Partner should use only one coordinate system when constructing the spatial area coverage for a collection or granule.

## Two-Dimensional Coordinate System

Granule-specific values can be searched against a two-dimensional coordinate system-based catalog search for Coordinate1 and Coordinate2. This search ability is not supported by all Data Providers.

Examples of two-dimensional coordinate system values are path/row for Worldwide Reference System (WRS) data and Moderate Resolution Imaging Spectroradiometer (MODIS) tiles IDs.

*Note: Two-dimensional coordinate system types are defined by ECHO operations and have to be enabled for a collection before it can be used in granule ingest.*

**Code Listing 17: Setting Two-Dimensional Coordinate System Coordinates**

```
<TwoDCoordinateSystem>
<StartCoordinate1>21</StartCoordinate1>
<StartCoordinate2>29</StartCoordinate2>
<EndCoordinate2>33</EndCoordinate2>
<TwoDCoordinateSystemName>WRS2</TwoDCoordinateSystemName>
</TwoDCoordinateSystem>
```

## Geometry Representations

Spatial data are most commonly described as geometry such as a polygon, a multi-polygon, or a line. They are stored as Oracle spatial objects in the database to record shape, spatial locations of corner points and spatial coordinate system used, Cartesian or Geodetic. Cartesian and Geodetic are considered different spatial representations. Corner points in the Cartesian system are connected by straight lines, while in the Geodetic system they are connected by great circle arcs. This makes their spatial coverage slightly different. Oracle requires specifying the coordinate system during data storage time. Therefore, data under different coordinate systems are also searched differently.

### Coordinate System

#### Cartesian Coordinate System

The Cartesian coordinate system is a flattened coordinate system with longitude ranged from –180 to 180 degrees and latitude ranged from –90 to 90 degrees. The projected map is flattened and open along the International Date Line with North Pole and South Pole as top and bottom line respectively.

#### Geodetic Coordinate System

The Geodetic coordinate system is defined in angular (latitude and longitude) and is defined relative to spherical polar coordinate and Earth Geodetic datum. Oracle defines coordinate system following OGC standards, which are defined at <http://www.opengeospatial.org/>. The Geodetic coordinate ECHO chose to support is

World Geodetic System 84 (WGS 84)—refer to Page 14 for the previous discussion of Geodetic coordinate system.

Also, refer to the table on Page 17 for a discussion supported spatial data type.

The following code sample shows how to define Geodetic coordinate system:

```
GEOGCS [ "Longitude / Latitude (WGS 84)", DATUM ["WGS 84",
SPHEROID ["WGS 84", 6378137.000000,
298.257224]], PRIMEM [ "Greenwich", 0.000000 ], UNIT ["Decimal
Degree", 0.01745329251994330]]
```

## Data Types and Representation

### *Point*

ECHO can receive and store spatial data representing one or more points. ECHO also supports searching for spatial data representing one or more points. In the XML metadata, follow the syntax shown in the following code sample to define a granule's spatial extent as a point:

**Code Listing 18: Single Point Example**

---

```
<SpatialDomainContainer>
  <HorizontalSpatialDomainContainer>
    <Point>
      <PointLongitude>-123.948</PointLongitude>
      <PointLatitude>45.0664</PointLatitude>
    </Point>
  </HorizontalSpatialDomainContainer>
</SpatialDomainContainer>
```

---

**Code Listing 19: Multiple Points**

---

```
<SpatialDomainContainer>
  <HorizontalSpatialDomainContainer>
    <Point>
      <PointLongitude>-123.948</PointLongitude>
      <PointLatitude>45.0664</PointLatitude>
    </Point>
    <Point>
      <PointLongitude>-133.546</PointLongitude>
      <PointLatitude>45.0664</PointLatitude>
    </Point>
  </HorizontalSpatialDomainContainer>
</SpatialDomainContainer>
```

---

## *Line*

ECHO can receive and store spatial data representing one or more lines. ECHO also supports searching for spatial data representing one or more lines. In the XML metadata, follow the syntax shown in the following code sample to define a granule's spatial extent as a line:

**Code Listing 20: Single Line Example**

---

```
<SpatialDomainContainer>
  <HorizontalSpatialDomainContainer>
    <Line>
      <Point>
        <PointLongitude>-123.948</PointLongitude>
        <PointLatitude>45.0664</PointLatitude>
      </Point>
      <Point>
        <PointLongitude>-133.546</PointLongitude>
        <PointLatitude>45.0664</PointLatitude>
      </Point>
    </Line>
  </HorizontalSpatialDomainContainer>
```

---

---

**Code Listing 21: Multiple Line Example**

---

```
<SpatialDomainContainer>
  <HorizontalSpatialDomainContainer>
    <Line>
      <Point>
        <PointLongitude>-123.948</PointLongitude>
        <PointLatitude>45.0664</PointLatitude>
      </Point>
      <Point>
        <PointLongitude>-133.546</PointLongitude>
        <PointLatitude>45.0664</PointLatitude>
      </Point>
    </Line>
    <Line>
      <Point>
        <PointLongitude>-123.948</PointLongitude>
        <PointLatitude>45.0664</PointLatitude>
      </Point>
      <Point>
        <PointLongitude>-133.546</PointLongitude>
        <PointLatitude>45.0664</PointLatitude>
      </Point>
      <Point>
        <PointLongitude>-143.546</PointLongitude>
        <PointLatitude>40.0664</PointLatitude>
      </Point>
    </Line>
  </HorizontalSpatialDomainContainer>
```

---

In the Cartesian coordinate system, the points are connected with a straight line on the plane in the order listed. The line connects two points, and the line will never cross the International Date Line or Poles.

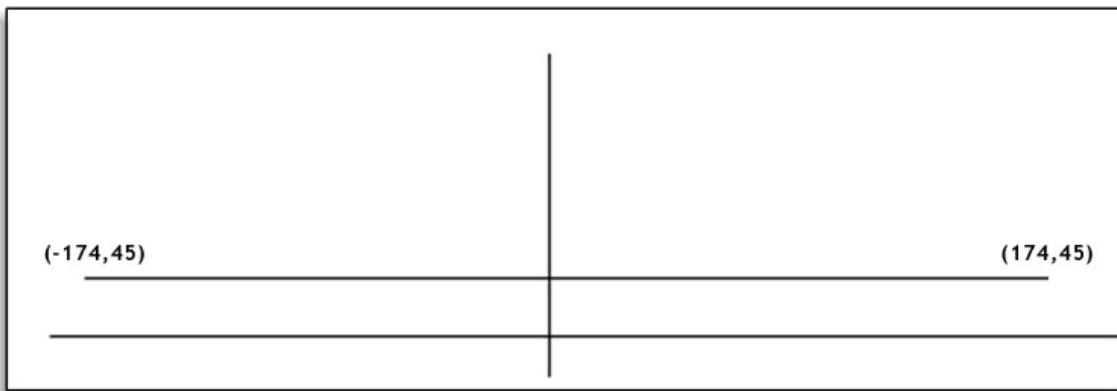
---

**Code Listing 22: Interpreted in Cartesian and Geodetic Systems**

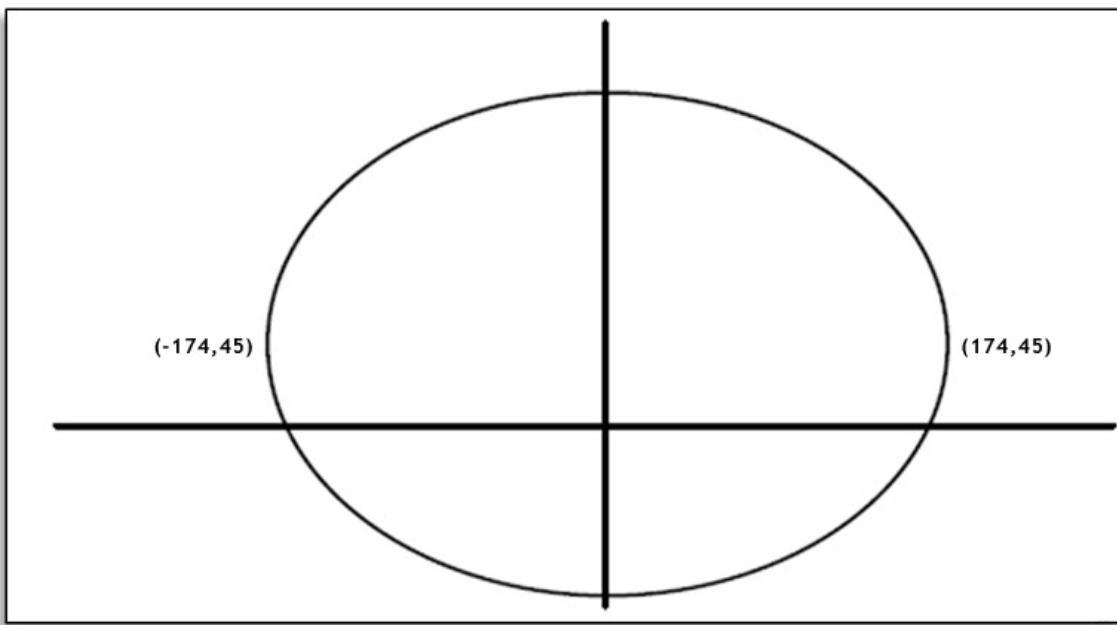
---

```
<SpatialDomainContainer>
  <HorizontalSpatialDomainContainer>
    <Line>
      <Point>
        <PointLongitude>-173.948</PointLongitude>
        <PointLatitude>45.0664</PointLatitude>
      </Point>
      <Point>
        <PointLongitude>173.546</PointLongitude>
        <PointLatitude>45.0664</PointLatitude>
      </Point>
    </Line>
  </HorizontalSpatialDomainContainer>
</SpatialDomainContainer>
```

The figure below shows a line in the Cartesian coordinate system:



This same expression in the Geodetic coordinate system represents the line as:



In the Geodetic coordinate system, the line connects two points using a great circle arc with the shortest distance between the two points. The line could cross the International Date Line and poles. If the same line indicated in the Cartesian coordinate system is represented in the Geodetic coordinate system, the code expression would be:

**Code Listing 23: Adding Density**

---

```
<SpatialDomainContainer>
    <HorizontalSpatialDomainContainer>
        <Line>
            <Point>
                <PointLongitude>-173.948</PointLongitude>
                <PointLatitude>45.0664</PointLatitude>
            </Point>
            <Point>
                <PointLongitude>0.0</PointLongitude>
                <PointLatitude>45.0664</PointLatitude>
            </Point>
            <Point>
                <PointLongitude>173.546</PointLongitude>
                <PointLatitude>45.0664</PointLatitude>
            </Point>
        </Line>
    </HorizontalSpatialDomainContainer>
</SpatialDomainContainer>
```

---

The additional points give the line more density so that Oracle interprets the data correctly. The appropriate density should be applied to the Geodetic coordinate system as well.

## Polygon

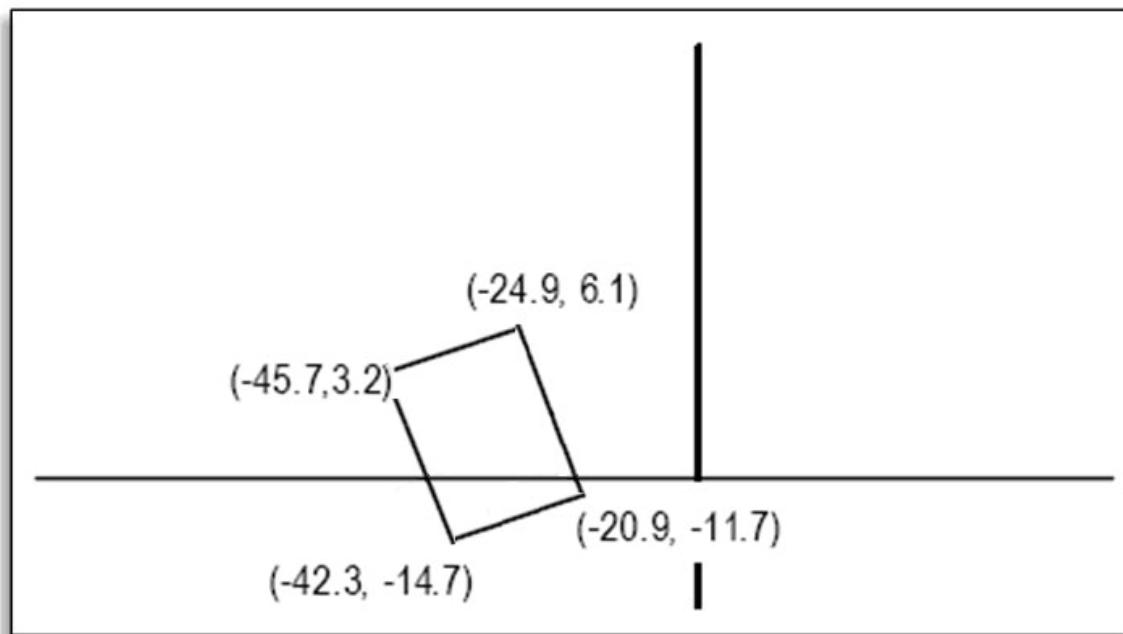
ECHO can receive and store spatial data representing a polygon, a polygon with hole, and multiple polygons (with or without holes). ECHO also supports searching for spatial data representing any of these polygons. In the XML metadata, follow the syntax shown in the following code sample to define a granule's spatial extent as a polygon:

**Code Listing 24: Single Polygon**

```
<SpatialDomainContainer>
  <HorizontalSpatialDomainContainer>
    <Polygon>
      <SinglePolygon>
        <OutRing>
          <Boundary>
            <Point>
              <PointLongitude>-20.9342</PointLongitude>
              <PointLatitude>-11.7045</PointLatitude>
            </Point>
            <Point>
              <PointLongitude>-42.3067</PointLongitude>
              <PointLatitude>-14.7732</PointLatitude>
            </Point>
            <Point>
              <PointLongitude>-45.7985</PointLongitude>
              <PointLatitude>3.198</PointLatitude>
            </Point>
            <Point>
              <PointLongitude>-24.8982</PointLongitude>
              <PointLatitude>6.1665</PointLatitude>
            </Point>
          </Boundary>
        </OutRing>
      </SinglePolygon>
    </Polygon>
  </HorizontalSpatialDomainContainer>
</SpatialDomainContainer>
```

A single polygon can have multiple holes, each represented by a single outer ring surrounding the area within it. In the Cartesian coordinate system, straight lines connect the points of the ring in the order in which they are listed, which must always be in clockwise order. The area should not cross the International Date Line or the poles. In the Geodetic coordinate system, the points are connected using a great circle arc according to the shortest distance between two points. Remember that a polygon coverage cannot span more than half the earth and may not cross the International Date Line and/or poles in the Geodetic coordinate system.

The figure below uses the Cartesian coordinate system to represent the spatial area covered when applying the code on the previous page:



---

**Code Listing 25: Single Polygon with a Hole**

---

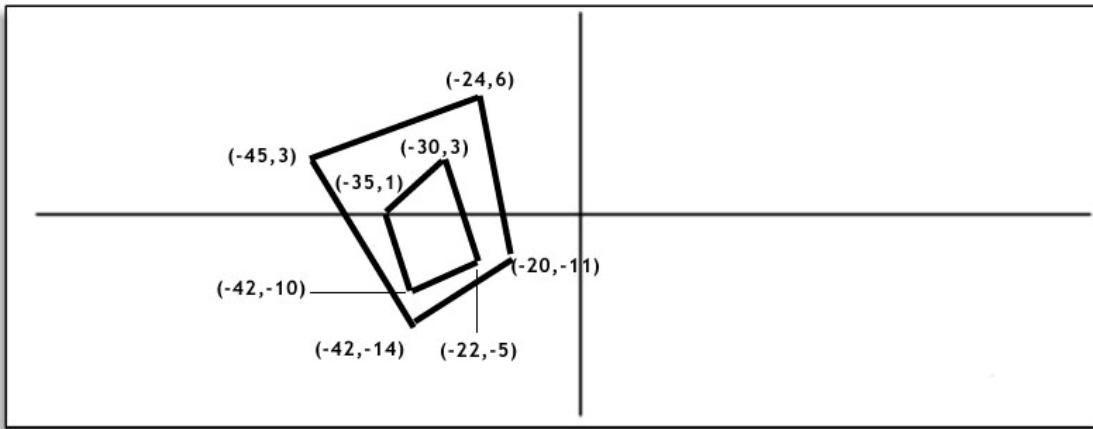
```
<SpatialDomainContainer>
  <HorizontalSpatialDomainContainer>
    <Polygon>
      <SinglePolygon>
        <OutRing>
          <Boundary>
            <Point>
              <PointLongitude>-19.9342</PointLongitude>
              <PointLatitude>-10.7045</PointLatitude>
            </Point>
            <Point>
              <PointLongitude>-42.3067</PointLongitude>
              <PointLatitude>-13.7732</PointLatitude>
            </Point>
            <Point>
              <PointLongitude>-44.7985</PointLongitude>
              <PointLatitude>3.198</PointLatitude>
            </Point>
            <Point>
              <PointLongitude>-23.8982</PointLongitude>
              <PointLatitude>6.1665</PointLatitude>
            </Point>
          </Boundary>
        </OutRing>
        <InnerRing>
          <Boundary>
            <Point>
              <PointLongitude>-21.9342</PointLongitude>
              <PointLatitude>-5.7045</PointLatitude>
            </Point>
            <Point>
              <PointLongitude>-42.3067</PointLongitude>
              <PointLatitude>-9.7732</PointLatitude>
            </Point>
            <Point>
              <PointLongitude>-34.7985</PointLongitude>
              <PointLatitude>1.198</PointLatitude>
            </Point>
            <Point>
              <PointLongitude>-29.8982</PointLongitude>
              <PointLatitude>3.1665</PointLatitude>
            </Point>
          </Boundary>
        </InnerRing>
      </SinglePolygon>
    </Polygon>
  </HorizontalSpatialDomainContainer>
</SpatialDomainContainer>
```

---

While a single polygon with a hole can have only one outer ring that represents the area surrounded within, it can have multiple inner rings that represent holes. All the rules,

restrictions and discussions for the outer ring in both coordinate systems apply to inner rings as well. An inner ring should be completely contained within the outer ring.

The figure below represents the spatial area covered when applying the code on the previous page:



### **Multiple Polygon**

A multiple polygon is a combination of polygons that may or may not have holes. Each single polygon expression should follow the same rules listed for polygon above. No two polygons may overlap each other.

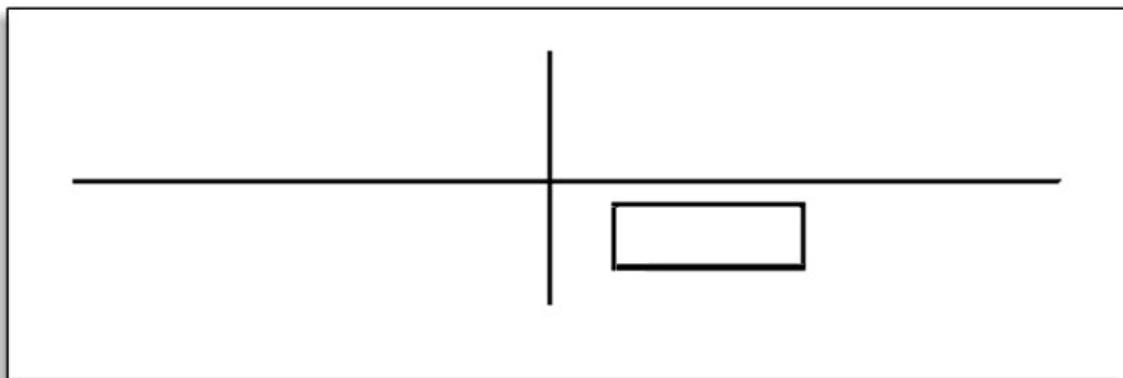
### **Bounding Box**

In the Cartesian coordinate system only, ECHO is capable of receiving, storing and supporting the search on spatial data representing a bounding box or multiple bounding boxes. To send ECHO spatial point data in the metadata XML file, follow the syntax shown in this code sample:

#### **Code Listing 26: Bounding Box**

```
<HorizontalSpatialDomainContainer>
  <BoundingRectangle>
    <WestBoundingCoordinate>8.733</WestBoundingCoordinate>
    <NorthBoundingCoordinate>-7.4861</NorthBoundingCoordinate>
    <EastBoundingCoordinate>43.199501</EastBoundingCoordinate>
    <SouthBoundingCoordinate>-35.2617</SouthBoundingCoordinate>
  </BoundingRectangle>
</HorizontalSpatialDomainContainer>
<HorizontalSpatialDomain>
  <ZoneIdentifier/>
  <BoundingRectangle>
    <WestBoundingCoordinate>8.733</WestBoundingCoordinate>
    <NorthBoundingCoordinate>-7.4861</NorthBoundingCoordinate>
    <EastBoundingCoordinate>43.199501</EastBoundingCoordinate>
    <SouthBoundingCoordinate>-35.2617</SouthBoundingCoordinate>
  </BoundingRectangle>
</HorizontalSpatialDomain>
```

The figure below represents the spatial area covered when applying the code shown above:



ECHO stores a bounding box as a four-pointed polygon, subject to the specifications and constraints described for the polygon. Bounding boxes cannot encompass more than half the earth and cannot cross the International Date Line or the poles. If there is more than one bounding box listed, then ECHO stores them as multiple polygons but requires that they not overlap.

## Invalid Spatial Representation

### Polygon Points in Counter-Clockwise Order

This spatial area expression is invalid in the Cartesian coordinate system. However, this same expression is considered valid in the Geodetic coordinate system, although the coverage will be interpreted differently.

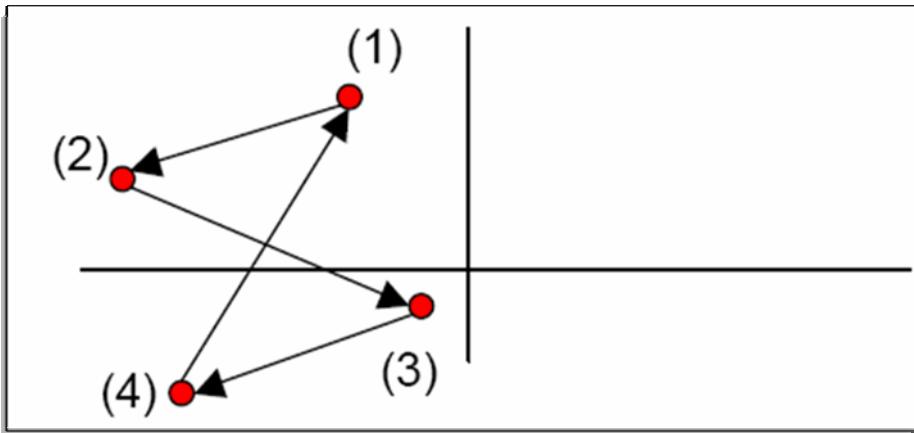
Code Listing 27: Polygon with Points in Counter-Clockwise Order

---

```
<SpatialDomainContainer>
  <HorizontalSpatialDomainContainer>
    <Polygon>
      <SinglePolygon>
        <OutRing>
          <Boundary>
            <Point>
              <PointLongitude>170</PointLongitude>
              <PointLatitude>30</PointLatitude>
            </Point>
            <Point>
              <PointLongitude>-170</PointLongitude>
              <PointLatitude>30</PointLatitude>
            </Point>
            <Point>
              <PointLongitude>-170</PointLongitude>
              <PointLatitude>-30</PointLatitude>
            </Point>
            <Point>
              <PointLongitude>170</PointLongitude>
              <PointLatitude>-30</PointLatitude>
            </Point>
          </Boundary>
        </OutRing>
      </SinglePolygon>
    </Polygon>
  </HorizontalSpatialDomainContainer>
</SpatialDomainContainer>
```

---

## Twisted Polygon

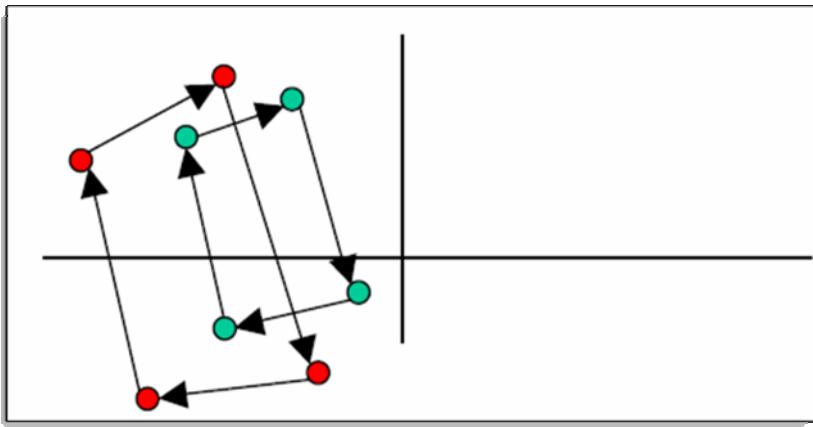


In either the Cartesian or Geodetic coordinate systems, the following code sample will result in an invalid polygon:

Code Listing 28: Twisted Polygon

```
<SpatialDomainContainer>
  <HorizontalSpatialDomainContainer>
    <Polygon>
      <SinglePolygon>
        <OutRing>
          <Boundary>
            <Point>
              <PointLongitude>-20.9342</PointLongitude>
              <PointLatitude>-11.7045</PointLatitude>
            </Point>
            <Point>
              <PointLongitude>-42.3067</PointLongitude>
              <PointLatitude>-14.7732</PointLatitude>
            </Point>
            <Point>
              <PointLongitude>-24.8982</PointLongitude>
              <PointLatitude>6.1665</PointLatitude>
            </Point>
            <Point>
              <PointLongitude>-45.7985</PointLongitude>
              <PointLatitude>3.198</PointLatitude>
            </Point>
          </Boundary>
        </OutRing>
      </SinglePolygon>
    </Polygon>
  </HorizontalSpatialDomainContainer>
</SpatialDomainContainer>
```

## Hole Crosses over Outer Ring



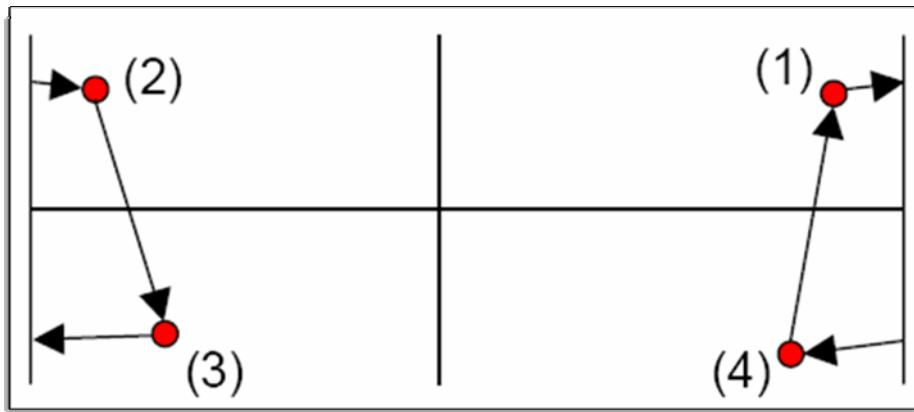
In either the Cartesian or Geodetic coordinate systems, the following code sample will result in an invalid data:

Code Listing 29: Hole Crosses over the Outer Ring

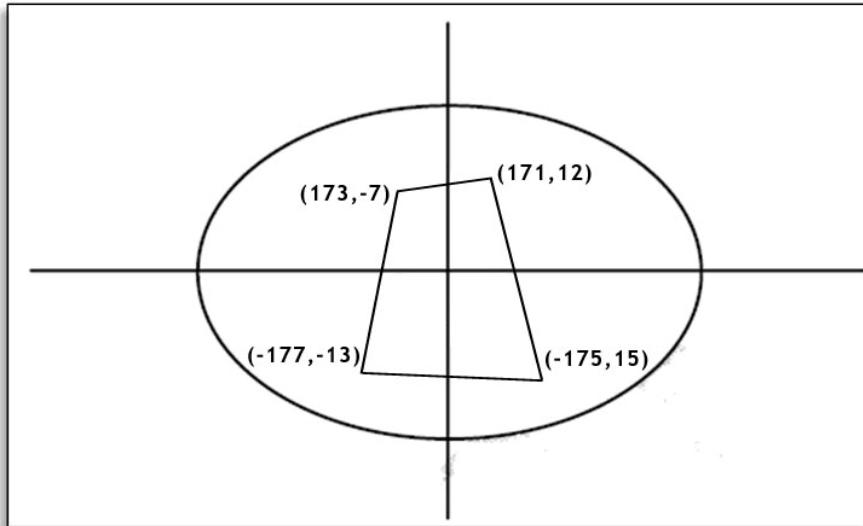
```
<SpatialDomainContainer>
  <HorizontalSpatialDomainContainer>
    <Polygon>
      <SinglePolygon>
        <OutRing>
          <Boundary>
            <Point>
              <PointLongitude>-20.9342</PointLongitude>
              <PointLatitude>-11.7045</PointLatitude>
            </Point>
            <Point>
              <PointLongitude>-42.3067</PointLongitude>
              <PointLatitude>-14.7732</PointLatitude>
            </Point>
            <Point>
              <PointLongitude>-45.7985</PointLongitude>
              <PointLatitude>3.198</PointLatitude>
            </Point>
            <Point>
              <PointLongitude>-24.8982</PointLongitude>
              <PointLatitude>6.1665</PointLatitude>
            </Point>
          </Boundary>
        </OutRing>
        <InnerRing>
          <Boundary>
            <Point>
              <PointLongitude>-17.9342</PointLongitude>
              <PointLatitude>-5.7045</PointLatitude>
            </Point>
            <Point>
              <PointLongitude>-30.3067</PointLongitude>
              <PointLatitude>-10.7732</PointLatitude>
            </Point>
          </Boundary>
        </InnerRing>
      </SinglePolygon>
    </HorizontalSpatialDomainContainer>
  </SpatialDomainContainer>
```

```
<Point>
    <PointLongitude>-35.7985</PointLongitude>
    <PointLatitude>1.198</PointLatitude>
</Point>
<Point>
    <PointLongitude>-10.8982</PointLongitude>
    <PointLatitude>3.1665</PointLatitude>
</Point>
</Boundary>
</InnerRing>
</SinglePolygon>
</Polygon>
</HorizontalSpatialDomainContainer>
</SpatialDomainContainer>
```

### Polygon Crosses International Date Line



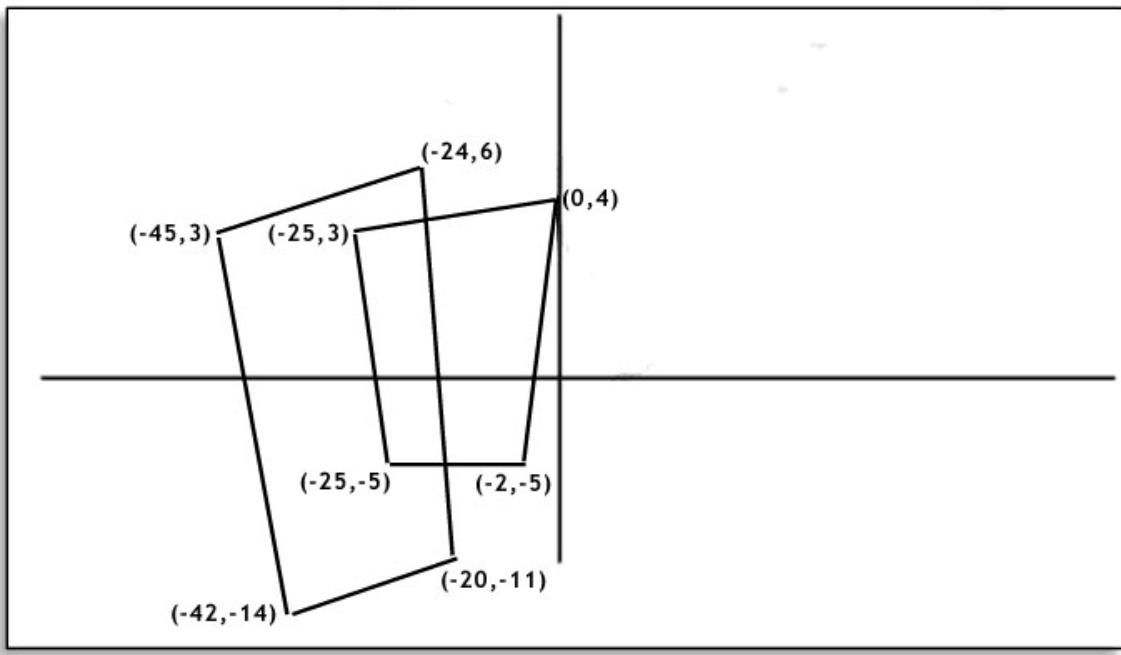
According to clockwise order, this expression in the Cartesian coordinate system represents a polygon crossing the International Date Line; therefore, it is invalid. However, in the Geodetic coordinate system, this is a valid spatial coverage area, as shown below:



**Code Listing 30: Polygon Crosses International Dateline**

```
<SpatialDomainContainer>
  <HorizontalSpatialDomainContainer>
    <Polygon>
      <SinglePolygon>
        <OutRing>
          <Boundary>
            <Point>
              <PointLongitude>170.9342</PointLongitude>
              <PointLatitude>11.7045</PointLatitude>
            </Point>
            <Point>
              <PointLongitude>-175.3067</PointLongitude>
              <PointLatitude>14.7732</PointLatitude>
            </Point>
            <Point>
              <PointLongitude>-176.7985</PointLongitude>
              <PointLatitude>-13.198</PointLatitude>
            </Point>
            <Point>
              <PointLongitude>172.8982</PointLongitude>
              <PointLatitude>-7.1665</PointLatitude>
            </Point>
          </Boundary>
        </OutRing>
      </SinglePolygon>
    </Polygon>
  </HorizontalSpatialDomainContainer>
</SpatialDomainContainer>
```

## Overlapping Polygons



In either coordinate system, this represents invalid spatial data.

**Code Listing 31: Overlapping Polygons**

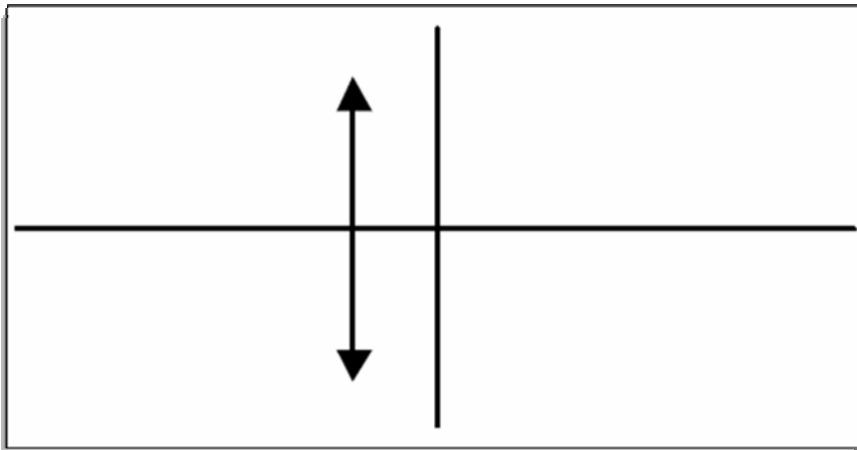
```
<SpatialDomainContainer>
  <HorizontalSpatialDomainContainer>
    <Polygon>
      <SinglePolygon>
        <OutRing>
          <Boundary>
            <Point>
              <PointLongitude>-19.9342</PointLongitude>
              <PointLatitude>-10.7045</PointLatitude>
            </Point>
            <Point>
              <PointLongitude>-42.3067</PointLongitude>
              <PointLatitude>-13.7732</PointLatitude>
            </Point>
            <Point>
              <PointLongitude>-44.7985</PointLongitude>
              <PointLatitude>3.198</PointLatitude>
            </Point>
            <Point>
              <PointLongitude>-23.8982</PointLongitude>
              <PointLatitude>6.1665</PointLatitude>
            </Point>
          </Boundary>
        </OutRing>
      </SinglePolygon>
    </Polygon>
```

---

```
<SinglePolygon>
  <OutRing>
    <Boundary>
      <Point>
        <PointLongitude>-1.9342</PointLongitude>
        <PointLatitude>-4.7045</PointLatitude>
      </Point>
      <Point>
        <PointLongitude>-25.3067</PointLongitude>
        <PointLatitude>-4.7732</PointLatitude>
      </Point>
      <Point>
        <PointLongitude>-24.7985</PointLongitude>
        <PointLatitude>3.198</PointLatitude>
      </Point>
      <Point>
        <PointLongitude>-0.2982</PointLongitude>
        <PointLatitude>4.1665</PointLatitude>
      </Point>
    </Boundary>
  </OutRing>
</SinglePolygon>
</Polygon>
</HorizontalSpatialDomainContainer>
</SpatialDomainContainer>
```

---

## Inappropriate Data



In this example, the bounding box is actually a line. It is important to use the correct spatial data type when constructing a spatial data expression. Do not use a bounding box to express a line.

*Note: The granularity of the spatial index for a given provider can result in two distinct but nearby points being interpreted as the same point.*

**Code Listing 32: Inappropriate Data**

```
<SpatialDomainContainer>
  <HorizontalSpatialDomainContainer>
    <BoundingRectangle>
      <WestBoundingCoordinate>-8.733</WestBoundingCoordinate>
      <NorthBoundingCoordinate>7.4861</NorthBoundingCoordinate>
      <EastBoundingCoordinate>-8.733</EastBoundingCoordinate>
      <SouthBoundingCoordinate>10.2617</SouthBoundingCoordinate>
    </BoundingRectangle>
  </HorizontalSpatialDomainContainer>
</SpatialDomainContainer>
```

---

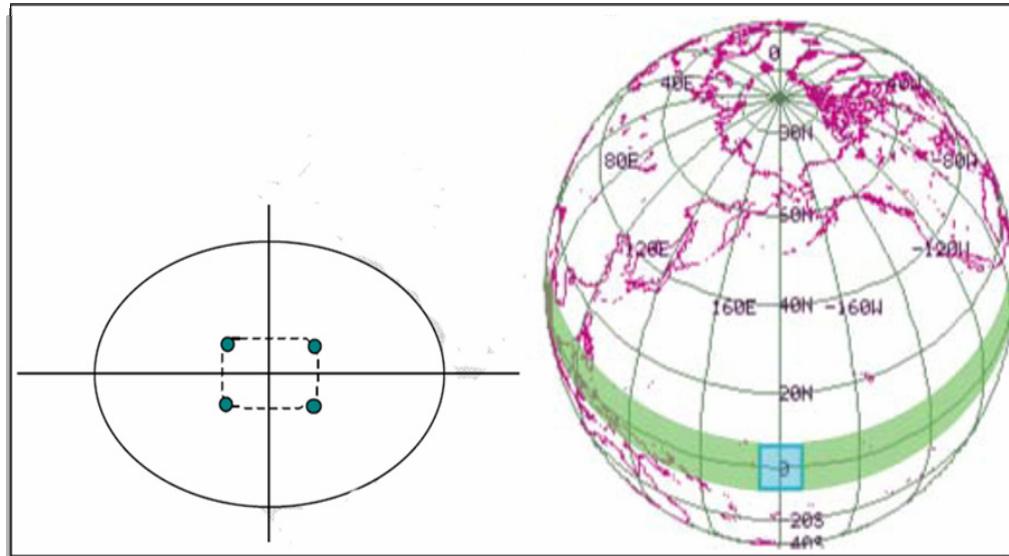
**Code Listing 33: Incorrect Density**

---

```
<SpatialDomainContainer>
<HorizontalSpatialDomainContainer>
<Polygon>
<SinglePolygon>
<OutRing>
<Boundary>
<Point>
    <PointLongitude>170.9342</PointLongitude>
    <PointLatitude>11.7045</PointLatitude>
</Point>
<Point>
    <PointLongitude>-175.3067</PointLongitude>
    <PointLatitude>14.7732</PointLatitude>
</Point>
<Point>
    <PointLongitude>-176.7985</PointLongitude>
    <PointLatitude>-13.198</PointLatitude>
</Point>
<Point>
    <PointLongitude>172.8982</PointLongitude>
    <PointLatitude>-7.1665</PointLatitude>
</Point>
</Boundary>
</OutRing>
</SinglePolygon>
</Polygon>
</HorizontalSpatialDomainContainer>
</SpatialDomainContainer>
```

---

The expression above is valid spatial data in the Geodetic coordinate system. However, the spatial coverage area represented will be as shown below:



You may want the large green band, but you will receive the small blue rectangle.

Oracle connects any two points using the shortest distance between points. To represent this spatial coverage correctly, you must increase the points' density by adding extra points. The sample below shows one way you might express these additional points, to represent this spatial coverage area correctly.

**Code Listing 34: Correct Density**

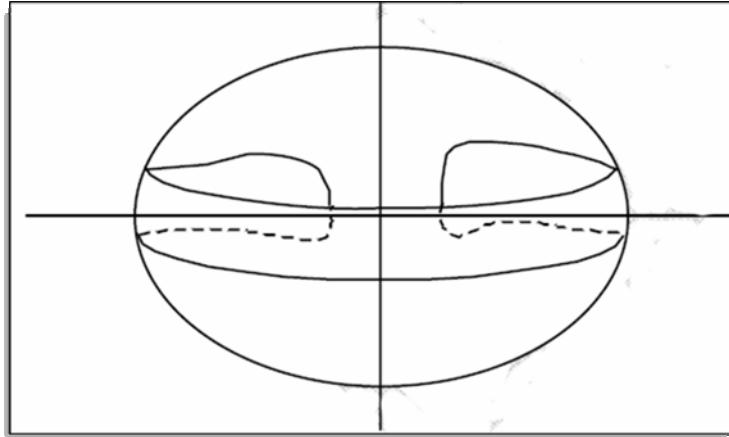
---

```
<SpatialDomainContainer>
  <HorizontalSpatialDomainContainer>
    <Polygon>
      <SinglePolygon>
        <OutRing>
          <Boundary>
            <Point>
              <PointLongitude>170.9342</PointLongitude>
              <PointLatitude>11.7045</PointLatitude>
            </Point>
            <Point>
              <PointLongitude>0.0</PointLongitude>
              <PointLatitude>14.7732</PointLatitude>
            </Point>
            <Point>
              <PointLongitude>-175.3067</PointLongitude>
              <PointLatitude>14.7732</PointLatitude>
            </Point>
            <Point>
              <PointLongitude>-176.7985</PointLongitude>
              <PointLatitude>-13.198</PointLatitude>
            </Point>
            <Point>
              <PointLongitude>0.0</PointLongitude>
              <PointLatitude>-10.1665</PointLatitude>
            </Point>
            <Point>
              <PointLongitude>172.8982</PointLongitude>
              <PointLatitude>-7.1665</PointLatitude>
            </Point>
          </Boundary>
          <OutRing>
        </SinglePolygon>
      </Polygon>
    </HorizontalSpatialDomainContainer>
  </SpatialDomainContainer>
```

---

## Incorrectly Defined Spatial Coverage

The spatial area coverage in this case is greater than half of the earth. This is invalid spatial data in both the Geodetic and the Cartesian coordinate systems because Oracle does not support spatial area greater than half of the earth in those two systems.



Code Listing 35: Polygon Covering More Than Half of the Earth

```
<SpatialDomainContainer>
  <HorizontalSpatialDomainContainer>
    <Polygon>
      <SinglePolygon>
        <OutRing>
          <Boundary>
            <Point>
              <PointLongitude>170.9342</PointLongitude>
              <PointLatitude>71.7045</PointLatitude>
            </Point>
            <Point>
              <PointLongitude>0.0</PointLongitude>
              <PointLatitude>71.7732</PointLatitude>
            </Point>
            <Point>
              <PointLongitude>-175.3067</PointLongitude>
              <PointLatitude>71.7732</PointLatitude>
            </Point>
            <Point>
              <PointLongitude>-176.7985</PointLongitude>
              <PointLatitude>-71.198</PointLatitude>
            </Point>
            <Point>
              <PointLongitude>0.0</PointLongitude>
              <PointLatitude>-71.1665</PointLatitude>
            </Point>
            <Point>
              <PointLongitude>172.8982</PointLongitude>
              <PointLatitude>-71.1665</PointLatitude>
            </Point>
          </Boundary>
        </SinglePolygon>
      </Polygon>
    </HorizontalSpatialDomainContainer>
  </SpatialDomainContainer>
```

---

```
</OutRing>
</SinglePolygon>
</Polygon>
</HorizontalSpatialDomainContainer>
</SpatialDomainContainer>
```

---

## Latitude/Longitude Range and Tolerance

ECHO makes use of resolution and range parameter settings to associate a level of precision and data range with spatial data. ECHO uses these parameters as evaluation parameters when validating spatial data input. Spatial data that use latitude/longitude are expressed in degrees, and tolerance is expressed in meters. For example, if the range for latitude is -50 to 50 and the range for longitude is -150 to 150, then ECHO will consider any granule input data with latitude and/or longitude falling outside of this range to be invalid. If the tolerance is 0.05 for both latitude and longitude, and if the distance between two points is less than 0.05 meter for both longitude and latitude, then those two points are considered the same point. In this situation, the spatial expression is invalid because ECHO spatial constructs require each point to have a unique spatial location. You should specifically define latitude/longitude range within your collection metadata.

ECHO defaults for range and tolerance are:

- Resolution: .1 meter (.0001 Km)
- Tolerance: .05 meter

## Online Data Access URL and Online Resources URL

For some granules or collections, the raw data are made available online via FTP or web URL. ECHO stores this online access information for directly accessible granule and collection data differently from information covering other aspects of granule and collection data. Directly accessible data require the <OnlineAccessURLs> tag and include the URL to that data.

Any other information covering aspects of the data, such as guides, product listings, validation information, etc., should be listed in a <CollectionOnlineResources> tag or <GranuleOnlineResources> tag, along with the URLs to that information.

## Ingest Reporting

The automated ingest process will generate an ingest summary report for you, with ingest start/stop time information, metadata files received with the file size respectively, number of collection/granule processed, etc. The ingest report will be sent to designated provider personnel within your organization (specified as ProviderContact when your organization originally registered with ECHO) via e-mail in XML format.

## New Items vs. Update Items

ECHO expects a complete set of granule or collection metadata for both inserts and updates. When processing a granule or collection, the ECHO system first checks to see if

the item already exists. If the item already exists in the system, ECHO will delete all the metadata associated with this item before inserting the new metadata in its place.

ECHO uses the item identification to search for the existing metadata that needs to be deleted before inserting new metadata. For a granule, the GranuleUR is assumed to be a unique identifier within your metadata. For a collection, the short name plus version number combined are assumed to be a unique identifier within your metadata. The ECHO system applies the same principle when dealing with the insertion or update of the granule or collection when processed against the XML metadata input. When updating a granule or collection, only the version with the most recent update time will be stored in the ECHO database. ECHO will ignore any granule or collection received that has a last update date that pre-dates the records already in the database.

## Metadata Ingest DTD

If you do not want to replace the complete record in ECHO with an update, ECHO allows you to specify fields to update individually. To update metadata, the provider has to follow ECHO's metadata update DTD and some other rules for the metadata update XML files.

Currently, ECHO handles a) granule level Online Access URL (delete, update, insert) and Measured Parameter QA Flags (update) metadata update and b) browse update (see section on browse below). ECHO ignores collection level metadata updates.

**Code Listing 36: Metadata Update DTD**

```
<?xml version="1.0" encoding="UTF-8"?>
<!ELEMENT ProviderService (UpdateMetadata)>
<!--UpdateMetadata can update a single collection, multiple
collections, a single granule, or multiple granules in one
transaction. Each update allows the addition of new metadata-->
<!ELEMENT UpdateMetadata (Collection*, Granule*)>
<!ELEMENT Collection (Target+, (Add | Update | Delete)+)>
<!ELEMENT Granule (Target+, (Add | Update | Delete)+)>
<!-- Target+ allows the same change to be made to several
different granules or collections simultaneously. This is
especially useful for bulk deletions of OnlineURLs. -->
<!ELEMENT Target (ID, ProviderLastUpdateDateTime,
SaveDateTimeFlag?)>
<!-- SaveDateTimeFlag is the flag that allows echo to update the
last update date time for Target. The default is SAVE -->
<!ELEMENT Add (QualifiedTag, MetadataValue)>
<!ELEMENT Update (QualifiedTag, MetadataValue)>
<!ELEMENT Delete (QualifiedTag+)>
<!ELEMENT QualifiedTag (#PCDATA)>
<!ELEMENT MetadataValue (#PCDATA)>
<!ELEMENT ProviderLastUpdateDateTime (#PCDATA)>
<!ELEMENT SaveDateTimeFlag (SAVE | DONTSAVE)>
<!ELEMENT SAVE EMPTY>
<!ELEMENT ID (#PCDATA)>
<!ELEMENT DONTSAVE EMPTY>
AdditionalAttributes element in Metadata Model
<!ELEMENT AdditionalAttributes (AdditionalAttribute)+>
<!ELEMENT AdditionalAttribute (AdditionalAttributeName,
AdditionalAttributeValue*, AdditionalAttributeValueType?)>
<!ELEMENT AdditionalAttributeName (#PCDATA)>
<!ELEMENT AdditionalAttributeValue (#PCDATA)>
<!ELEMENT AdditionalAttributeValueType (#PCDATA)>
```

The **QualifiedTag** is a string that follows the XPath standard. (XPath is a language for finding elements information in an XML document.) It is used to indicate which item in the DTD representing ECHO's metadata model is to be manipulated (for example, updated, inserted, or deleted). In the case of deleting all online URLs for granules or collections, the XPath need only look like “**OnlineAccessURLs**”. An optional flag, **SaveDateTimeFlag**, allows you to change the date/time timestamp that identifies when metadata was last updated, to reflect new update date/time information.

## Delete Items

Instruct ECHO to delete collections or granules by placing the collections or granules under the <DeleteCollections> or <DeleteGranules> tag, respectively. ECHO uses only the item's identification to process the deletion of the item and all the metadata associated with this item. ECHO will keep the deleted items' identification and deletion date in the database for metadata history auditing purposes. The only way to re-install the collections or granules in the ECHO system is to re-submit the metadata for those items to ECHO for insertion.

## Data Not Included in the DTD

The DTD does not provide descriptions of certain data characteristics such as data type, length of field, etc. This additional information is available in the ECHO data dictionary located on the ECHO Web site at <http://www.echo.nasa.gov/reference/reference.shtml>. If data received does not meet the specifications described in the data dictionary, those data records will be ignored and could cause ingest of the file containing these data records to fail.

For any repetitive item identification in the metadata input, ECHO will ingest into the database only one item bearing the most recent update date with its complete information. The rest of the items and their associated information will be ignored even if they contain different data information and will result in a duplicate error in the ingest summary for that item.

## Values for the Qualified Tag for Updates, Deletes, and Inserts

The following DTD excerpts list the acceptable tags for indicating which field in the ECHO data model to update.

*Note: Only the following fields in the data model are available for partial update: OnlineResourceURL, OnlineAccessURL, and ScienceQualityField.*

Code Listing 36: DTD Excerpt for Collection Metadata

```
<!ELEMENT OnlineAccessURLs (OnlineAccessURL +)>
<!ELEMENT OnlineAccessURL (URL, URLDescription?,MimeType)>
<!ELEMENT CollectionOnlineResources (OnlineResource +)>
<!ELEMENT OnlineResource (OnlineResourceURL,
OnlineResourceDescription?, OnlineResourceType,
OnlineResourceMimeType)>
<!ELEMENT URL (#PCDATA)>
<!ELEMENT URLDescription (#PCDATA)>
<!ELEMENT MimeType (#PCDATA)>
<!ELEMENT OnlineResourceURL (#PCDATA)>
<!ELEMENT OnlineResourceDescription (#PCDATA)>
<!ELEMENT OnlineResourceType (#PCDATA)>
<!ELEMENT OnlineResourceMimeType (#PCDATA)>
```

---

**Code Listing 37: DTD Excerpt for Granule Metadata**

---

```
<!ELEMENT OnlineAccessURLs (OnlineAccessURL +)>
<!ELEMENT OnlineAccessURL (URL, URLDescription?,MimeType)>
<!ELEMENT GranuleOnlineResources (OnlineResource +)>
<!ELEMENT OnlineResource (OnlineResourceURL,
OnlineResourceDescription?, OnlineResourceType,
OnlineResourceMimeType)>
<!ELEMENT URL (#PCDATA)>
<!ELEMENT URLDescription (#PCDATA)>
<!ELEMENT MimeType (#PCDATA)>
<!ELEMENT OnlineResourceURL (#PCDATA)>
<!ELEMENT OnlineResourceDescription (#PCDATA)>
<!ELEMENT OnlineResourceType (#PCDATA)>
<!ELEMENT OnlineResourceMimeType (#PCDATA)>
```

---

Use `OnlineAccessURLs` only for the actual data. For URLs to all other kinds of Web pages (including metadata), send these as `OnlineResources`.

This table identifies valid XPath values for the qualified tags.

Tag	Function	XPath	Value
Granule URL	Delete One	OnlineAccessURLs/OnlineAccessURL[URL="old_url"]/URL	
Granule URL	Delete All	OnlineAccessURLs	
Granule URL	Update	OnlineAccessURLs /OnlineAccessURL[URL="old_url"]/URL	New URL
Granule URL	Insert	OnlineAccessURLs /OnlineAccessURL/URL url	URL
Granule Online Resource	Delete One	GranuleOnlineResources/OnlineResource[OnlineResourceURL="old_url"]/ OnlineResourceURL	
Online Resource URL	Update	GranuleOnlineResources/OnlineResource[OnlineResourceURL="old_url"] /OnlineResourceURL	New URL
Granule Online Resource Type	Update	GranuleOnlineResources/OnlineResource[OnlineResourceURL ="old_url"]/OnlineResourceType	New type
Granule Online Resource Mime Type	Insert	GranuleOnlineResources/OnlineResource[OnlineResourceURL ="old_url"]/OnlineResourceMimeType	Mime type
Granule Automatic QA Flag	Update	MeasuredParameter/MeasuredParameterContainer [ParameterName="p1"]/QAFlags/AutomaticQualityFlag	New flag
Granule Automatic QA Flag	Update	MeasuredParameter/MeasuredParameterContainer [ParameterName="p1"]/QAFlags/AutomaticQualityFlagExplanation	New expl.

Tag	Function	XPath	Value
Granule Operational QA Flag	Update	MeasuredParameter/MeasuredParameterContainer [ParameterName="p1"]/QAFlags/OperationalQualityFlag	New flag
Granule Operational QA Flag Explanation	Update	MeasuredParameter/MeasuredParameterContainer [ParameterName="p1"]/QAFlags/OperationalQualityFlagExplanation	New expl.
Granule Science QA Flag	Update	MeasuredParameter/MeasuredParameterContainer [ParameterName="p1"]/QAFlags/ScienceQualityFlag	New flag
Granule Science QA Flag Explanation	Update	MeasuredParameter/MeasuredParameterContainer [ParameterName="p1"]/QAFlags/ScienceQualityFlagExplanation	New expl.
Additional Attribute	Delete One	AdditionalAttributes/AdditionalAttribute[AdditionalAttributeName= "old_name"]/AdditionalAttributeName	
Additional Attributes	Delete All	AdditionalAttributes	
Additional Attributes	Update	AdditionalAttributes/AdditionalAttribute[AdditionalAttributeName= "old_name"]/AdditionalAttributeName  AdditionalAttributes/AdditionalAttribute[AdditionalAttributeName= "old_name"]/AdditionalAttributeValue  AdditionalAttributes/AdditionalAttribute[AdditionalAttributeName= "old_name"]/AdditionalAttributeValueType	New name  New value  New type
Additional Attributes	Insert	AdditionalAttributes/AdditionalAttribute/AdditionalAttributeName  AdditionalAttributes/AdditionalAttribute[AdditionalAttributeName= "name"]/AdditionalAttributeValue  AdditionalAttributes/AdditionalAttribute[AdditionalAttributeName= "name"]/AdditionalAttributeValueType	name  value  type

## Examples

### Partial Metadata Insert

---

```
<?xml version="1.0" encoding="UTF-8"?>
<ProviderAccountService>
    <UpdateMetadata>
        <Granule>
            <Target>
                <ID>SC:MIL2TCAL.002:17872867</ID>
                <ProviderLastUpdateDateTime>2007-04-12
00:02:46.95</ProviderLastUpdateDateTime>
                <SaveDateTimeFlag><DONTSAVE/></SaveDateTimeFlag>
            </Target>
            <Add>
                <QualifiedTag>GranuleOnlineResources/OnlineResource/OnlineResourc
eURL</QualifiedTag>
                <MetadataValue>ftp://10dps01u.ecs.nasa.gov/distribution/BRWS/Brow
se.001/2007.04.03/MISBR.A2007093.0137.005.2007101133056.AN.24.hdf
            </MetadataValue>
            </Add>
            <Add>
                <QualifiedTag>GranuleOnlineResources/OnlineResource[OnlineResourc
eURL="ftp://10dps01u.ecs.nasa.gov/distribution/BRWS/Browse.001/20
07.04.03/MISBR.A2007093.0137.005.2007101133056.AN.24.hdf"]/Online
ResourceType</QualifiedTag>
                <MetadataValue>BROWSE</MetadataValue>
            </Add>
            <Add>
                <QualifiedTag>GranuleOnlineResources/OnlineResource[OnlineResourc
eURL="ftp://10dps01u.ecs.nasa.gov/distribution/BRWS/Browse.001/20
07.04.03/MISBR.A2007093.0137.005.2007101133056.AN.24.hdf"]/Online
ResourceMimeType</QualifiedTag>
                <MetadataValue>image/jpeg</MetadataValue>
            </Add>
            <Add>
                <QualifiedTag>OnlineAccessURLs/OnlineAccessURL/URL</QualifiedTag>
                <MetadataValue>ftp://10dps01u.ecs.nasa.gov/longterm/MISR/MIL2TCAL
.002/2007.04.03/MISR_AM1_TC_ALBEDO_P114_0038776_F04_0008.hdf</Met
adataValue>
            </Add>
            <Add>
                <QualifiedTag>OnlineAccessURLs/OnlineAccessURL[URL="ftp://10dps01
u.ecs.nasa.gov/longterm/MISR/MIL2TCAL.002/2007.04.03/MISR_AM1_TC_
ALBEDO_P114_0038776_F04_0008.hdf"]/MimeType</QualifiedTag>
                    <MetadataValue>application/x-
hdfeos</MetadataValue>
            </Add>
            <Add>
                <QualifiedTag>GranuleOnlineResources/OnlineResource/OnlineResourc
eURL</QualifiedTag>
```

---

---

```
<MetadataValue>ftp://10dps01u.ecs.nasa.gov/longterm/MISR/MIL2TCAL
.002/2007.04.03/MISR_AM1_TC_ALBEDO_P114_0038776_F04_0008.hdf.xml<
/MetadataValue>
    </Add>
    <Add>
<QualifiedTag>GranuleOnlineResources/OnlineResource[OnlineResourc
eURL="ftp://10dps01u.ecs.nasa.gov/longterm/MISR/MIL2TCAL.002/2007
.04.03/MISR_AM1_TC_ALBEDO_P114_0038776_F04_0008.hdf.xml"]/OnlineR
esourceType</QualifiedTag>
    <MetadataValue>METADATA</MetadataValue>
    </Add>
    <Add>
<QualifiedTag>GranuleOnlineResources/OnlineResource[OnlineResourc
eURL="ftp://10dps01u.ecs.nasa.gov/longterm/MISR/MIL2TCAL.002/2007
.04.03/MISR_AM1_TC_ALBEDO_P114_0038776_F04_0008.hdf.xml"]/OnlineR
esourceMimeType</QualifiedTag>
    <MetadataValue>text/xml</MetadataValue>
    </Add>
</Granule>
</UpdateMetadata>
</ProviderAccountService>
```

---

### Partial Metadata Delete

```
<?xml version="1.0" encoding="UTF-8"?>
<ProviderAccountService>
    <UpdateMetadata>
        <Granule>
            <Target>
                <ID>SC:MIL3DAE.004:17176039</ID>
                <ProviderLastUpdateTime>2007/04/23
08:32:56</ProviderLastUpdateTime>
                <SaveDateTimeFlag><DONTSAVE/></SaveDateTimeFlag>
            </Target>
            <Target>
                <ID>SC:MIL3DLS.004:17176041</ID>
                <ProviderLastUpdateTime>2007/04/23
08:32:56</ProviderLastUpdateTime>
                <SaveDateTimeFlag><DONTSAVE/></SaveDateTimeFlag>
            </Target>
            <Target>
                <ID>SC:MIL3DAE.004:17176328</ID>
                <ProviderLastUpdateTime>2007/04/23
08:32:56</ProviderLastUpdateTime>
                <SaveDateTimeFlag><DONTSAVE/></SaveDateTimeFlag>
            </Target>
            <Target>
                <ID>SC:MIL3DLS.004:17176329</ID>
                <ProviderLastUpdateTime>2007/04/23
08:32:56</ProviderLastUpdateTime>
                <SaveDateTimeFlag><DONTSAVE/></SaveDateTimeFlag>
            </Target>
            <Delete>
```

---

```
        <QualifiedTag>OnlineAccessURLs</QualifiedTag>
<QualifiedTag>GranuleOnlineResources</QualifiedTag>
    </Delete>
</Granule>
</UpdateMetadata>
</ProviderAccountService>
```

---

### **Partial Granule Update**

---

```
<?xml version="1.0" encoding="UTF-8"?>
<ProviderAccountService>
<UpdateMetadata>
<Granule>
    <Target>
        <ID>SC:MOD14.004:2034404150</ID>
        <ProviderLastUpdateTime>2007-04-13
12:07:09.603</ProviderLastUpdateTime>
    </Target>
    <Update>
        <QualifiedTag>MeasuredParameter/MeasuredParameterContainer[Parame
terName="MODIS L2 Active Fire
Detection"]/QAFlags/ScienceQualityFlag</QualifiedTag>
            <MetadataValue>Being Investigated</MetadataValue>
        </Update>
    </Granule>
    <Granule>
        <Target>
            <ID>SC:MOD14.004:2034404150</ID>
            <ProviderLastUpdateTime>2007-04-13
12:07:09.603</ProviderLastUpdateTime>
        </Target>
        <Update>
            <QualifiedTag>MeasuredParameter/MeasuredParameterContainer[Parame
terName="MODIS L2 Active Fire
Detection"]/QAFlags/ScienceQualityFlagExplanation</QualifiedTag>
                <MetadataValue>Product assessment is
ongoing</MetadataValue>
            </Update>
        </Granule>
    </UpdateMetadata>
</ProviderAccountService>
```

---

## Browse Image Files and Browse Metadata

Browse files provide a high-level view of granule or collection data. If you have Browse, you should send both Browse image files and a Browse metadata XML files to the ECHO system. The ECHO system will allocate the storage for Browse files, build a Browse image URL, and update the database to associate the Browse URL to its item record.

When ECHO processes the Browse ingest, all the Browse image files indicated by the tag <InternalFileName> under the tag of <BrowseCrossReference> in the Browse metadata input XML file must be included as well. In addition to checking for the existence of the Browse image files, ECHO also verifies the actual Browse image file size against the file size indicated for that file in the Browse metadata input XML file. If any Browse image file indicated in the Browse metadata input XML file does not exist, or if any file size does not match the indicated size, then the Browse image files and Browse metadata input XML file will not be processed.

After processing, ECHO places the Browse image files online and associates them with the appropriate granule(s). This information is made available to the end user. The file name in the <InternalFileName> tag is the provider's unique identifier of the Browse file. ECHO supports many-to-many referencing between granules and Browse files.

---

**Code Listing 39: Browse Metadata Input XML File**

---

```
<?xml version="1.0" encoding="UTF-8"?>
<BrowseReferenceFile>
<DTDVersion>1.0</DTDVersion>
<DataCenterId>EDC</DataCenterId>
<TemporalCoverage>
<StartDate>2002-10-17T00:00:00.000Z</StartDate>
<EndDate>2002-10-18T00:00:00.000Z</EndDate>
</TemporalCoverage>
<BrowseCrossReference>
<GranuleUR>SC:L70RWRS.002:2008440693</GranuleUR>
<BrowseGranuleId></BrowseGranuleId>
<InsertTime></InsertTime>
<LastUpdate>2002-10-17 18:55:33.996</LastUpdate>
<InternalFileName>:BR:Browse.001:2008440612:1.BINARY</InternalFil
eName>
<BrowseSize>167554.0</BrowseSize>
</BrowseCrossReference>
<BrowseCrossReference>
<GranuleUR>SC:L70RWRS.002:2008440702</GranuleUR>
<BrowseGranuleId></BrowseGranuleId>
<InsertTime></InsertTime>
<LastUpdate>2002-10-17 18:56:04.216</LastUpdate>
<InternalFileName>:BR:Browse.001:2008440624:1.BINARY</InternalFil
eName>
<BrowseSize>159823.0</BrowseSize>
</BrowseCrossReference>
<BrowseCrossReference>
<GranuleUR>SC:L70RWRS.002:2008440732</GranuleUR>
<BrowseGranuleId></BrowseGranuleId>
<InsertTime></InsertTime>
<LastUpdate>2002-10-17 18:56:14.856</LastUpdate>
<InternalFileName>:BR:Browse.001:2008440630:1.BINARY</InternalFil
eName>
<BrowseSize>65258.0</BrowseSize>
</BrowseCrossReference>
</BrowseReferenceFile>
```

---

## Browse Insert, Update, Replace and Delete

ECHO allows you to update/replace and delete Browse references. Collection and granule metadata associated with the Browse metadata to be ingested must be in the ECHO database.

Browse metadata for ingest must conform to the Browse DTD as shown:

### Browse DTD

Code Listing 40: Sample Browse XML File

```
<?xml version="1.0" encoding="UTF-8"?>
<!--DTD generated by XMLSpy v2005 rel. 3 U
(http://www.altova.com)-->
<!ELEMENT BrowseReferenceFile (DTDVersion, DataCenterId,
TemporalCoverage, DeleteBrowse*, BrowseCrossReference*)>
<!ELEMENT DTDVersion (#PCDATA)>
<!ELEMENT DataCenterId (#PCDATA)>
<!ELEMENT TemporalCoverage (StartDate, EndDate)>
<!ELEMENT StartDate (#PCDATA)>
<!ELEMENT EndDate (#PCDATA)>
<!ELEMENT DeleteBrowse (((ShortName, VersionID) | DataSetID |
GranuleUR), InternalFileName*, BrowseCollectionId?,
BrowseGranuleId?)>
<!ATTLIST DeleteBrowse
SilentDeleteErrorHandling NMTOKEN "0">
<!ELEMENT ShortName (#PCDATA)>
<!ELEMENT VersionID (#PCDATA)>
<!ELEMENT DataSetID (#PCDATA)>
<!ELEMENT GranuleUR (#PCDATA)>
<!ELEMENT InternalFileName (#PCDATA)>
<!ELEMENT BrowseCollectionId (#PCDATA)>
<!ELEMENT BrowseGranuleId (#PCDATA)>
<!ELEMENT BrowseCrossReference (((ShortName, VersionID) |
DataSetID | GranuleUR), BrowseCollectionId?, BrowseGranuleId?,
InsertTime?, LastUpdate?, DeleteTime?, InternalFileName,
BrowseDescription?, BrowseSize?)>
<!ELEMENT InsertTime (#PCDATA)>
<!ELEMENT LastUpdate (#PCDATA)>
<!ELEMENT DeleteTime (#PCDATA)>
<!ELEMENT BrowseDescription (#PCDATA)>
<!ELEMENT BrowseSize (#PCDATA)>
```

The following table defines the elements used in Browse ingest insert/update/delete:

**Table 3: Browse Elements**

Element	Definition
DTDVersion	This element specifies the version of the Browse DTD, for example, 9.0. Currently, this field is not captured by ingest.
DataCenterId	The DataCenterID of the data center from which the ingest is originating. Note: this field is not captured by ingest.
GranuleUR	The GranuleUR for the granule associated with the Browse file.
StartDate	Temporal start date of the data. Note: Currently is not captured by ingest.
EndDate	Temporal end date of the data. Note: currently is not captured by ingest.
BrowseCrossReference	The tag needed for inserting or updating Browse files
DeleteBrowse	The tag needed for deleting Browse files
InternalFileName	The name of the Browse image you will be associating with the granule.
InsertTime	Insert time that the granule was inserted into ECHO.
LastUpdate	The LocalLastUpdate time for the granule in ECHO. The new LastUpdate date must be more recent than old LastUpdate date when updating Browse.
DeleteTime	The time to delete the Browse from the system. If DeleteTime tag not used, then deletion occurs instantaneously.
BrowseSize	The size of the Browse file being inserted or updated for a granule. The value in the Browse ingest file MUST match the actual file size.
SilentDeleteErrorHandler	=0: will return error message if any needed item (e.g. ShortName, VersionID, DatasetID, or GranuleUR) does not exist. =1: turns off error messages

The following DTDs can also be downloaded from the ECHO website at <http://www.echo.nasa.gov/reference/reference.shtml> and are listed in “Appendix D: Ingest DTDs,” Page 107.

### ***Insert Browse Example***

The Code Listing below is an XML example of inserting Browse references into multiple granules. The sample XML shows inserting one sample Browse image file into a single granule. This sample will insert a unique Browse image file into each granule.

**Code Listing 41: Sample Browse XML File**

---

```
<?xml version="1.0" encoding="UTF-8"?>
<BrowseReferenceFile>
<DTDVersion>1.0</DTDVersion>
<DataCenterId>EDC</DataCenterId>
<TemporalCoverage>
<StartDate>2005-05-12T00:00:00.000Z</StartDate>
<EndDate>2005-06-12T00:00:00.000Z</EndDate>
</TemporalCoverage>
<BrowseCrossReference>
<GranuleUR>10x10ReferenceGranule0:0</GranuleUR>
<LastUpdate>2005-05-12 07:13:57.05</LastUpdate>
<InternalFileName>test.hdf0</InternalFileName>
<BrowseSize>10</BrowseSize>
</BrowseCrossReference>
<BrowseCrossReference>
<GranuleUR>10x10ReferenceGranule0:1</GranuleUR>
<LastUpdate>2005-05-12 06:47:12.383</LastUpdate>
<InternalFileName>test.hdf1</InternalFileName>
<BrowseSize>11</BrowseSize>
</BrowseCrossReference>
<BrowseCrossReference>
<GranuleUR>10x10ReferenceGranule0:2</GranuleUR>
<LastUpdate>2005-05-12 06:30:03.576</LastUpdate>
<InternalFileName>test.hdf2</InternalFileName>
<BrowseSize>12</BrowseSize>
</BrowseCrossReference>
<BrowseCrossReference>
<GranuleUR>10x10ReferenceGranule0:3</GranuleUR>
<LastUpdate>2005-05-12 06:22:19.323</LastUpdate>
<InternalFileName>test.hdf3</InternalFileName>
<BrowseSize>13</BrowseSize>
</BrowseCrossReference>
<BrowseCrossReference>
<GranuleUR>10x10ReferenceGranule0:4</GranuleUR>
<LastUpdate>2005-05-12 05:23:35.37</LastUpdate>
<InternalFileName>test.hdf4</InternalFileName>
<BrowseSize>14</BrowseSize>
</BrowseCrossReference>
</BrowseReferenceFile>
```

---

To submit the Browse metadata for ingest, copy the file and the corresponding Browse image files to the /provider/data/browse FTP location for your provider.

## ***Update Browse Example***

The following code listing is an example of updating Browse references in multiple granules. The sample XML shows updating the Browse references for the five granules shown in the previous code listing.

**Code Listing 42: Update Browse Example: testbrowseu.xml**

---

```
<?xml version="1.0" encoding="UTF-8"?>
<BrowseReferenceFile>
    <DTDVersion>1.0</DTDVersion>
    <DataCenterId>EDC</DataCenterId>
    <TemporalCoverage>
        <StartDate>2005-05-12T00:00:00.000Z</StartDate>
        <EndDate>2005-06-12T00:00:00.000Z</EndDate>
    </TemporalCoverage>
    <BrowseCrossReference>
        <GranuleUR>10x10ReferenceGranule0:0</GranuleUR>
        <LastUpdate>2005-05-16 07:13:57.05</LastUpdate>
        <InternalFileName>test.hdf1</InternalFileName>
        <BrowseSize>11</BrowseSize>
    </BrowseCrossReference>
    <BrowseCrossReference>
        <GranuleUR>10x10ReferenceGranule0:1</GranuleUR>
        <LastUpdate>2005-05-16 06:47:12.383</LastUpdate>
        <InternalFileName>test.hdf2</InternalFileName>
        <BrowseSize>12</BrowseSize>
    </BrowseCrossReference>
    <BrowseCrossReference>
        <GranuleUR>10x10ReferenceGranule0:2</GranuleUR>
        <LastUpdate>2005-05-16 06:30:03.576</LastUpdate>
        <InternalFileName>test.hdf3</InternalFileName>
        <BrowseSize>13</BrowseSize>
    </BrowseCrossReference>
    <BrowseCrossReference>
        <GranuleUR>10x10ReferenceGranule0:3</GranuleUR>
        <LastUpdate>2005-05-16 06:22:19.323</LastUpdate>
        <InternalFileName>test.hdf4</InternalFileName>
        <BrowseSize>14</BrowseSize>
    </BrowseCrossReference>
    <BrowseCrossReference>
        <GranuleUR>10x10ReferenceGranule0:4</GranuleUR>
        <LastUpdate>2005-05-16 05:23:35.37</LastUpdate>
        <InternalFileName>test.hdf0</InternalFileName>
        <BrowseSize>10</BrowseSize>
    </BrowseCrossReference>
</BrowseReferenceFile>
```

---

*NOTE: For the Browse updates to be successful, the file name must be the same as the insert filename.*

*Example: If you inserted a Browse image into a granule, and your ingest file was named testbrowse.xml, then your update file must also be named testbrowse.xml. If the file names are not the same, the Browse function will treat the ingest file as an insert instead of an update.*

To submit an update Browse request, copy the Browse update file and the corresponding browse image files to the /provider/data/browse FTP location for your provider.

### **Delete Browse Example**

The following code listing is an example of deleting Browse references. The sample XML shows the removal of the Browse references previously inserted and updated in the two previous code listings.

**Code Listing 43: Delete Browse Example: testbrowsed.xml**

```
<?xml version="1.0" encoding="UTF-8"?>
<BrowseReferenceFile>
    <DTDVersion>1.0</DTDVersion>
    <DataCenterId>EDC</DataCenterId>
    <TemporalCoverage>
        <StartDate>2005-04-11T00:00:00.000Z</StartDate>
        <EndDate>2005-04-12T00:00:00.000Z</EndDate>
    </TemporalCoverage>
    <DeleteBrowse SilentDeleteErrorHandler = "0">
        <GranuleUR>10x10ReferenceGranule0:0</GranuleUR>
        <InternalFileName>test.hdf1</InternalFileName>
    </DeleteBrowse>
    <DeleteBrowse SilentDeleteErrorHandler = "0">
        <GranuleUR>10x10ReferenceGranule0:1</GranuleUR>
        <InternalFileName>test.hdf2</InternalFileName>
    </DeleteBrowse>
    <DeleteBrowse SilentDeleteErrorHandler = "0">
        <GranuleUR>10x10ReferenceGranule0:2</GranuleUR>
        <InternalFileName>test.hdf3</InternalFileName>
    </DeleteBrowse>
    <DeleteBrowse SilentDeleteErrorHandler = "0">
        <GranuleUR>10x10ReferenceGranule0:3</GranuleUR>
        <InternalFileName>test.hdf4</InternalFileName>
    </DeleteBrowse>
    <DeleteBrowse SilentDeleteErrorHandler = "0">
        <GranuleUR>10x10ReferenceGranule0:4</GranuleUR>
        <InternalFileName>test.hdf0</InternalFileName>
    </DeleteBrowse>
</BrowseReferenceFile>
```

To delete browse metadata from ECHO, copy a Browse delete file and the corresponding Browse image files to the /provider/data/browse ftp location for your provider.

## Error Messages

During preprocessing, ECHO checks your input files for errors unrelated to ingest. Then ECHO processes ingest metadata and sends you an ingest summary report via e-mail upon successful ingest. This report, presented in XML, describes any abnormalities discovered in the input file.

### ***Input File Validation Error Messages***

Before ECHO ingests your metadata, it analyzes and validates them against its corresponding DTD. If it detects an error, ECHO rejects the input file and sends an input file error report to the e-mail address registered in your ECHO user account. The following is an example of an input file error report.

**Code Listing 44: Example of the Input File Error Report**

---

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- ECHO Ingest Input File Error Report: Wed Mar 19 10:20:01 EST
2003 -->
<!-- created by ECHO Ingest System -->
<InputFileErrorSummary>
<DataProvider>GCC</DataProvider>
<ProcessStartTime>03/19/2003 10:20:01</ProcessStartTime>
<CollectionInputFileWithError>
<XMLValidationError>
<FileName>VTCCLSR72001101659031903.XML</FileName>
<ErrorMessage>
file read took 0.005 seconds
error parsing: Error: Content model for CollectionAssociation
does not allow it to end here
in unnamed entity at line 245 char 26 of [unknown]
</ErrorMessage>
</XMLValidationError>
<NonXMLFile>VTCCLSR72001101659031903.XML.met</NonXMLFile>
</CollectionInputFileWithError>
<ProcessStopTime>03/19/2003 10:20:02</ProcessStopTime>
</InputFileErrorSummary>
```

---

- Non-XML File

If the input file does not start with the standard XML file declaration line such as <?xml ...>, then this input file is considered a “non-XML” file and is rejected for the ingest process. The input file error summary report will list the error with an error message like the one shown below:

---

```
<NonXMLFile>VTCCLSR72001101659031903.XML.met</NonXMLFile>
```

---

- XML File Put in Wrong Entry Directory

ECHO creates a directory under your FTP home directory for specific types of input XML files. For instance, all collection metadata XML files might go to a .../collection directory, while all granule XML files might go to a .../granule directory. If the XML file that is deposited in the .../collection directory is not a collection XML file, ECHO will reject this file for ingest processing. The input file error summary report will list the error message listed as:

---

```
<Non...XMLFile>VTCGMOLT200110920011240101.XML</Non...XMLFile>
```

---

The “...” specifies a category such as “Collection,” “Granule,” or “Browse” depending on the category of data being examined.

- Invalid XML File

ECHO will validate each input XML file against its corresponding DTD. If for any reason the DTD validation fails, ECHO will move this file to your backup space for further investigation. The input file error summary report will list the error with an error message like the one shown below:

**Code Listing 45: XML Validation Error**

---

```
<XMLValidationError>
<FileName>VTCCLSR72001101659031903.XML</FileName>
<ErrorMessage>
file read took 0.005 seconds
error parsing:
Error: Content model for CollectionAssociation does not allow it
to end here in unnamed entity at line 245 char 26 of
[unknown]
</ErrorMessage>
</XMLValidationError>
```

---

The text in the tag ErrorMessage is generated by the XML validation utility, which usually stops validation as soon as it detects the first error.

### **Data Ingest Error Messages**

When your metadata has been successfully ingested, you will receive an ingest summary report via the e-mail address in your ECHO user account.

---

**Code Listing 46: Example of the Ingest Summary Report for a Successful Ingest**

---

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- ECHO Ingest Summary Report: Wed Nov 13 13:55:01 EST 2002 -->
<!-- created by ECHO Ingest System -->
<IngestSummary>
  <DataProvider>ETE_TEST</DataProvider>
  <IngestStartTime>11/13/2002 13:55:01</IngestStartTime>
  <Collection>
    <InputFiles>
      <File name="VTCCLSR7200211130011240555.XML" size="8424" />
    </InputFiles>
    <Processed total="1">
      <Replacement total="1">
        <DataSets>
          <DataSet shortname="L7CPF" version="2" />
        </DataSets>
        </Replacement>
        <Processed>
        </Collection>
        <Granule>
          <InputFiles>
            <File name="VTCGLSR7200211130011240101042555.XML" size="1967" />
          </InputFiles>
          <Processed total="2">
            <Insertion total="1" />
            <Replacement total="1" />
          </Processed>
        </Granule>
      <IngestStopTime>11/13/2002 13:56:26</IngestStopTime>
    </IngestSummary>
```

---

There are five types of ingest information:

- <Browse>
- <Valids>
- <Collection>
- <Granule>
- <Updates>

For Browse and Valid, only the input file name and size are listed to show which input files ECHO has processed. For Collection, Granule, and Updates, more detailed error information is given. The detailed ingest process information is listed under the <Processed> section of the summary report. Under the <Processed> element, various tags will appear indicating a transaction or an error. The total number of collections or granules that fall into the transaction or error tag is listed as the tag's attribute. For a collection, a list of dataset short names and version numbers will be reported for each tag that indicates a transaction or error. ECHO breaks a large input file into small chunks to process one at a time. The total granules in a <Process> tag will not be larger than 1,000. If it is larger than 1,000 granules, ECHO Operations uses scripts to separate the process into smaller chunks. You can use the <Process> element multiple times for an input file.

The possible error messages associated with each transaction within the Ingest Summary Report appear below.

*Note: A collection or granule that produces any of the errors listed below will not be ingested.*

- Out-of-Date Update for Collection or Granule

If the date of a collection or granule in the ECHO database is more recent than the date on the same collection or granule in your input, ECHO will return an error and not process the specific record. In the example below,  $n$  represents the total number of collections or granules in your input, and  $m$  represents how many of  $n$  are out of date (that is, how many pre-date the same collections or granules in the database). If the error occurs at the granule level, the report will reflect only the total number, as represented by  $n$ .

**Code Listing 47: Error Returned When Input Data Is Older Than the Information Recorded in the ECHO Database**

```
<Processed total="n">
.....
<OutOfDate total="m">
<DataSets>
<DataSet shortname="collection's short name" version="version
number" />
.....
</DataSets>
</OutOfDate>
.....
</Processed>
```

- Duplicated Input

More than one input collection with the same identifier in the same round of ingest will return the sample error shown below. When such duplicates occur, ECHO ingests the first one in the file bearing the newest collection and ignores any other duplicate. In the example shown below,  $n$  represents the total number of collections or granules in your input, and  $m$  represents how many of  $n$  are duplicates. If the error occurs at the granule level, the report will reflect only the total number, as represented by  $n$ .

**Code Listing 48: Error Returned Because of Duplicate Identifiers in a Single Round of Ingest**

---

```
<Processed total=" $n$ ">
.....
<Duplicated total=" $m$ ">
<DataSets>
<DataSet shortname="collection's short name" version="version
number" />
.....
</DataSets>
</Duplicated>
.....
</Processed>
```

---

- Invalid Spatial Data

Invalid spatial data will return the sample error shown below. In the example shown below,  $n$  represents the total number of collections or granules in your input, and  $m$  represents how many of  $n$  contain invalid spatial data. If the error occurs at the granule level, the report will reflect only the total number, as represented by  $n$ .

For information on spatial rules for avoiding invalid spatial errors, refer to Page 36, the “Spatial Representations, Coordinates and Projections” section in “Chapter 6: Creating Order Options.”

**Code Listing 49: Error Returned When the Spatial Coverage Area Data for a Collection or Granule Is Invalid**

---

```
<Processed total=" $n$ ">
.....
<InvalidSpatial total=" $m$ ">
<DataSets>
<DataSet shortname="collection's short name" version="version
number" />
.....
</DataSets>
</InvalidSpatial>
.....
</Processed>
```

---

- Data That Violates Data Constraints

ECHO will reject a granule or collection record with bad data, such as a number attribute that contains a character string. ECHO will return an error like the sample shown below,

where  $n$  represents the total number of collections or granules in your input, and  $m$  represents how many of  $n$  contain bad data. If the error occurs at the granule level, the report will reflect only the total number, as represented by  $n$ .

If the data that violates the data constraint appears within the collection's required information, then none of the collection will be ingested into the database. If you notice that the total number of collections processed, as represented by  $n$ , is less than the total number you submitted, contact the ECHO Operations team for investigation.

**Code Listing 50: Error Returned When Any Piece of Data Associated with a Collection or Granule Has Violated a Data Constraint**

---

```
<Processed total="n">
.....
<RecordWithBadData total="m">
<DataSets>
<DataSet shortname="collection's short name" version="version
number" />
.....
</DataSets>
</RecordWithBadData>
.....
</Processed>
```

---

- Invalid Deletion

If your input contains an identifier for metadata that you want to delete, but the identifier does not exist in the database, ECHO will return an error like the sample shown below, where  $n$  represents the total number of collections or granules in your input, and  $m$  represents how many of  $n$  are deletions with non-existent identifiers. If the error occurs at the granule level, the report will reflect only the total number, as represented by  $n$ .

**Code Listing 51: Error Returned When a Collection or Granule Identifier Sent for Deletion Does Not Exist in the ECHO Database**

---

```
<Processed total="n">
...
<InvalidDeletion total="m">
<DataSets>
<DataSet shortname="collection's short name" version="version
number" />
.....
</DataSets>
</InvalidDeletion>
.....
</Processed>
```

---

- Granule Associated with a Collection that Does Not Exist in the ECHO Database

An input granule that references a collection that does not exist in the ECHO database will return the sample error shown below, where  $n$  represents the total number of collections or granules in your input, and  $m$  represents how many of  $n$  are granules referencing non-existent collections. This circumstance may occur when the collection being referenced is part of the same round of input but has failed be ingested. If the error occurs at the granule level, the report will reflect only the total number, as represented by  $n$ .

**Code Listing 52: Error Returned Because Collection Referenced by Input Granule(s) Does Not Exist in the ECHO Database**

---

```
<Processed total="n">
..
<AssociatedCollectionNotExist total="m">
..
</Processed>
```

---

- Metadata Update Errors

The error messages below will be returned when the ingest process encounters metadata update errors:

- “Metadata update tags do not exist or are incorrect”
- “The granule does not exist”
- “The new ProviderLastUpdateDateTime is earlier than the current ProviderLastUpdateDateTime in ECHO”
- “Attempting to update with no MetadataValue provided”
- “Attempting to delete or update non-exist data URL”
- “Attempting to insert existing data URL”
- “Attempting to delete or update non-exist online resource URL”
- “Attempting to insert existing online resource URL”
- “QA Flag allows update only”
- “Attempting to update QA Flag with incorrect parameter name”
- “Missing online resource type during online resource URL insert”

Code Listings 58-60 show metadata update error messages.

**Code Listing 53: Non-XML File Format**

---

```
<MetadataUpdateInputFileWithError>
<NonXMLFile>file name</NonXMLFile>
</MetadataUpdateInputFileWithError>
```

---

---

**Code Listing 54: Non-MetadataUpdate XML File**

---

```
<MetadataUpdateInputFileWithError>
<NonMetadataUpdateXMLFile>file name</ NonMetadataUpdateXMLFile>
</MetadataUpdateInputFileWithError>
```

---

**Code Listing 55: Invalid Metadata Update**

---

```
<MetadataUpdateInputFileWithError>
<XMLValidationError>
<FileName>metadataupdate.xml</FileName>
<ErrorMessage>xxx</ErrorMessage>
</XMLValidationError>
</MetadataUpdateInputFileWithError>
```

---

***Date Error Ignored by ECHO***

When the information associated with date is not expressed in a format that you previously set up with ECHO Operations, the date information will be ignored. The collection or granule will be ingested in the database with partially incorrect data (missing dates).

For example, you notified ECHO Operations that you will use the date format “yyyy-mm-dd hh24:mi:ss” (such as “2002-03-15 14:50:00”), but the date information in your input file looks like the following:

---

```
<LastUpdate>2002-3-15 2:50:00PM</LastUpdate>
```

---

Because your date format is inconsistent with the date format that you set up with ECHO Operations, ECHO will store the collection or granule but will not store the LastUpdateDate.

# Chapter 4: Validating Your Metadata

## View Dataset Information

The GetDatasetInformation call gives granule-level details in the given dataset. The granules must match the given criteria. This method is similar to performing a query; however, it is highly optimized for Data Partner reconciliation of metadata.

Due to the large amount of data that may be returned by this method, FTP Push is the only supported delivery mechanism.

You may specify temporal ranges to limit the search. If specified, the range type field in the granule must be between the start and stop times.

The available online parameters will only return granules if they are available online. If you set **temporalRanges** to false, then ECHO will return all matching granules.

All of the restriction fields (dataset ID, ranges, and online flag) will be joined together with the Boolean AND when the search is performed. The standard FTP URL format is:

`ftp://[user ID:password]host_name[:port]/[path name/][file name]`

To ensure uniqueness of the file name, ECHO generates a new unique name.

The following describes the web service call:

---

```
< GetDatasetInformation>
```

---

### Parameters:

The parameters are defined below:

**Token:** The security token

**dataSetId:** The ID of the dataset from where the information comes

**temporalRanges:** The temporal ranges to apply when searching

**availableOnline:** This should be set to true to restrict results to granules available online, or to false to get all granules regardless of online status.

**browseAvailable:** This should be set to true to restrict the results to granules which have browse data available online, or false to get all granules regardless of browse data availability.

**ftpUrl:** The FTP location to push the results to

### Returns:

The unique filename that will be created on the FTP server as the results are processed

## Code Example

---

```
<s0:GetDatasetInformation
  xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:xs="http://www.w3.org/2001/XMLSchema"
  xmlns:echo="http://echo.nasa.gov/echo/v9/types"
  xmlns:s0="http://echo.nasa.gov/echo/v9">

  <s0:token>TOKEN_ABCDEFG12345</s0:token>

  <s0:datasetId>MODIS_EXAMPLE_V001</s0:datasetId>

  <s0:temporalRanges>

    <echo:Item>

      <echo:StartDate>2006-04-
      13T23:45:00.000Z</echo:StartDate>

      <echo:StopDate>2007-04-
      13T16:19:04.921Z</echo:StopDate>

    </echo:Item>

  </s0:temporalRanges>

  <s0:availableOnline>true</s0:availableOnline>

  <s0:browseAvailable>true</s0:browseAvailable>

  <s0:ftpUrl>ftp://user:mypassword@myserver:21/echo/result</s0:ftpU
  rl>

</s0:GetDatasetInformation>
```

---

# Chapter 5: Controlling Access to Your Metadata

## Managing Data Access Rules

Use the Provider User Management Program (PUMP) or the API itself to set up Data Access Rules as discussed in this chapter. To download PUMP and its user's guide, go to [http://www.echo.nasa.gov/data\\_partners/data\\_tools.shtml](http://www.echo.nasa.gov/data_partners/data_tools.shtml).

You can manage access to your metadata using the Data Management Service by creating **DataAccessRules** that restrict or permit ECHO users to take actions on the data.

Supported **Actions** include Viewing, Browsing, and Ordering metadata.

### Data Access Rules

**DataAccessRules** can be applied to specific groups of ECHO users to give data access permission to users who have similar access privileges. You may use groups only for permission data access rules. If you have not specified any groups in **DataAccessRules**, then the rules apply to every ECHO user.

A set of **DataValue** values in a **DataAccessRule** specifies which collections or granules the rule covers. A single **DataAccessRule** can only apply to collections or granules, not both. If the rule is for collections, then **DataValue** should contain dataset IDs. If the rule is for granules, then **DataValue** should contain granule URs (Universal Reference). If you have not specified any value, then the rule applies to both collections and granules.

Conditions can further restrict which collections or granules the data rule covers. There are four types of conditions:

- Temporal
- Rolling Temporal
- Restriction Flag
- Boolean

Use the **ConditionComparator** in the data access rule to compare the condition values with the metadata value.

**Temporal** and **Rolling Temporal** conditions are based on time fields in the metadata. **Temporal** specifies a start and end time. The fields **Production date**, **Acquisition date**, **Insert date**, and **Last Update date** can be used for Temporal conditions but only with the equals and not equals comparators.

**Rolling Temporal** specifies an amount of time after the current time and uses the same fields as Temporal. Rolling Temporal can only be used with the greater than, greater than or equal to, less than, and less than or equal to comparators.

**Restriction Flag** indicates that the data access rule should be based on the value of the restriction flag in the metadata. The field “Lower” in the Restriction Flag condition indicates a value for the restriction flag field in the metadata. Restriction Flag conditions can be used with any comparator.

**Boolean** is a very basic condition that you should use if you do not want one of the other conditions. It has a single Boolean flag, and if the flag is true, then the data access rule will be applied. If the flag is false, then the data access rule will not be applied. Setting the flag to false is equivalent to not creating a data access rule. Boolean conditions can only be used with the equals comparator.

Examples:

---

```
<BooleanConditionField>
    <ConditionName>True condition</ConditionName>
    <Description>This is a true data condition. False condition
use BooleanFlag set to be false.</Description>
    <BooleanFlag>true</BooleanFlag>
</BooleanConditionField>

<TemporalConditionField>
    <ConditionName>year 2006</ConditionName>
    <Description>This is year 2006 data. Use StartTime and StopTime
for the correct range. Use TargetTemporalField to specify
temporal type. Possible enums are PRODUCTION, ACQUISITION,
INSERT, LAST_UPDATE.</Description>
    <StartTime>2006-01-01T00:00:00.00Z</StartTime>
    <StopTime>2006-12-31T11:59:59.00Z</StopTime>
    <TargetTemporalField><PRODUCTION/></TargetTemporalField>
</TemporalConditionField>

<RollingTemporalConditionField>
    <ConditionName>30 days duration</ConditionName>
    <Description>This is 30 days duration condition. Use Duration
to specify duration in milliseconds: 30x24x60*60*100 =
259200000.</Description>
    <Duration>259200000</Duration>
    <TargetTemporalField><PRODUCTION/></TargetTemporalField>
</RollingTemporalConditionField>

<RestrictionFlagConditionField>
    <ConditionName>Restriction flag lower than 3</ConditionName>
    <Description>This condition is used to restrict access to data
whose restriction flag is lower than 3</Description>
    <Lower>3</Lower>
</RestrictionFlagConditionField>
```

---

# Chapter 6: Creating Order Options

Most Data Partners use the Provider User Management Program (PUMP), available at [http://www.echo.nasa.gov/data\\_partners/data\\_tools.shtml](http://www.echo.nasa.gov/data_partners/data_tools.shtml), to create order options, which allow you to configure a block of custom provider policies that will define the information that a client must provide as part of an order, as well as other settings such as communications and firewall rules. Refer to the next chapter for details on communication configuration.

## Order Options

The option hierarchy in ECHO was created to be as flexible as possible in terms of Data Partner ordering needs. For details about ECHO Order Fulfillment, refer to [http://www.echo.nasa.gov/data\\_partners/data\\_tools.shtml](http://www.echo.nasa.gov/data_partners/data_tools.shtml).

*Note: Not all catalog items have associated definitions – in which case they do not need option selections when ordering. However, if there are any options that are required by that catalog item, they will have to be filled out before you can validate and/or submit the order (or else an error will be returned).*

Although you can create order options in PUMP, the output will be in ECHO Forms. For details about ECHO Forms, refer to the ECHO Forms schema and other information at [http://www.echo.nasa.gov/data\\_partners/data\\_tools.shtml](http://www.echo.nasa.gov/data_partners/data_tools.shtml).

### Assigning Option Definitions to Catalog Options

#### OptionDefinitions

Option Definitions allow third parties to define parts of the API on ECHO using XML. Option Selections contain XML that must conform to the Option Definitions.

#### Elements

Option assignment calls take a list of option definitions described below. Refer to this list of the major elements used in Option Definitions, which shows either a short definition of the element or the web page that defines them in detail.

**Guid:** (Multiplicity: 0 to 1) string

<<http://api.echo.nasa.gov/echo/ws/v9/string.html>>

**Name:** (Multiplicity: 1 to 1) This is the name of the definition. This must be unique per provider.

### **OptionScope:** <<http://api.echo.nasa.gov/echo/ws/v9/OptionScope.html>>

**Scope:** (Multiplicity: 1 to 1) Use the option form Scope to determine if the option is a system-level option or a provider level option. Only administrators may add system-level definitions and only Data Providers may add provider-level definitions.

**Deprecated:** (Multiplicity: 0 to 1) Indicates if the definition is deprecated (obsolete). Deprecated definitions will be returned to the client and will be considered valid when validating an order; however, new order items cannot be added using the definition and existing order items cannot be updated using the definition. This flag is ignored when creating an option definition; however, it will always be set and returned once the definition has been created in ECHO.

### **AnyElement:** <<http://api.echo.nasa.gov/echo/ws/v9/AnyElement.html>>

**Form:** (Multiplicity: 1 to 1) The ECHO Form itself. Contents must conform to ECHO Forms schema. Consult [http://www.echo.nasa.gov/data\\_partners/data\\_tools.shtml](http://www.echo.nasa.gov/data_partners/data_tools.shtml) for more information.

### Calls

These operations require Data Provider or Administrator access. They add the given option definitions to the pool of order options for provider. Use order options to collect provider-specific information for orders. You can read more about the calls at <http://api.echo.nasa.gov/echo/ws/v9/index.html>.

### Faults

#### **InternalFault:** <<http://api.echo.nasa.gov/echo/ws/v9/InternalFault.html>>

#### **AuthorizationFault:** <<http://api.echo.nasa.gov/echo/ws/v9/AuthorizationFault.html>>

#### **ItemNotFoundFault:**

<<http://api.echo.nasa.gov/echo/ws/v9/ItemNotFoundFault.html>>

#### **InvalidArgumentException:**

<<http://api.echo.nasa.gov/echo/ws/v9/InvalidArgumentException.html>>

#### **DuplicateIdFault:** <<http://api.echo.nasa.gov/echo/ws/v9/DuplicateIdFault.html>>

#### **InvalidStateFault:** <<http://api.echo.nasa.gov/echo/ws/v9/InvalidStateFault.html>>

#### **ParseFault:** <<http://api.echo.nasa.gov/echo/ws/v9/ParseFault.html>> ,

#### **ValidationFault:** <<http://api.echo.nasa.gov/echo/ws/v9/ValidationFault.html>>

### Parameters

**token:** The Security Token with Provider Role or administrator

**optionDefinitions:** The option definitions to be added.—at least one is required.

## Returns

ECHO returns list of GUIDs identifying the created option definitions.

The GUID assigned is a unique filename that will be created on the FTP server as the results are processed.

**This page is intentionally left blank.**

# **Chapter 7: Setting Up Communication Configurations for Orders**

## **General Information for Configuration**

Most Data Partners use the Provider User Management Program (PUMP) to configure communications for orders. To download PUMP and its user's guide, go to [http://www.echo.nasa.gov/data\\_partners/data\\_tools.shtml](http://www.echo.nasa.gov/data_partners/data_tools.shtml).

## **What Must Be Specified**

### **Provider Policies**

Establish a set of Data Provider policies for orders. Each provider has a set of provider policies that contain information such as the endpoint for the provider (or adapter), the number of attempts to complete the transaction, which operations are supported, and the amount of time between attempts. These definitions are standardized to easy-to-understand business objects.

### ***Endpoint for Receiving Orders***

You must specify an “end point,” a Uniform Resource Identifier (URI), also known as network address, where ECHO can send the order. The network address is usually either an IP or HTTP address. Within PUMP, select Provider Policies and type the URI into the End Point text box (under routing).

You need to make the exact URI location available for each of the user actions that you support. This location is where ECHO ultimately sends the actual transaction message.

### ***Secure Socket Layer (SSL)***

The Order Fulfillment API fully supports SSL encrypted order transmissions because ECHO supports certificate authority (CA) signed certificates, such as Verisign certificates, as well as self-signed certificates.

Set the certificate information. The basic steps in configuration of an SSL are:

1. The certificate must be generated or purchased.
2. The server hosting your endpoint must be configured to use SSL.
3. The provider policies must be configured with the public, PEM-encoded key from the certificate.

For more information and Order Options Samples (XML), refer to [http://www.echo.nasa.gov/data\\_partners/data\\_tools.shtml](http://www.echo.nasa.gov/data_partners/data_tools.shtml).

**This page is intentionally left blank.**

# Chapter 8: Receiving and Fulfilling Orders

The following information refers to standard, relevant fields in most Data Partner's Order Options that you, as a Data Partner, will read when you receive an order. This information is generic. Most Data Partners would consider these standard options; however, your operation may use different options. For that reason, the examples shown after the descriptions of each standard field do not necessarily correspond to your own configuration.

## Order Fulfillment

For Order Fulfillment API Documentation, Order Fulfillment Types XML Schema, and the Order Fulfillment Service WSDL, refer to  
[http://www.echo.nasa.gov/data\\_partners/data\\_tools.shtml](http://www.echo.nasa.gov/data_partners/data_tools.shtml).

## Order Identifier

The OrderId provides identifying information about an order. The ProviderId is ECHO's name for you, the Data Partner. The OrderId is ECHO's unique identifier (a GUID) for the order. The TrackingId is where your unique order tracking number is maintained for future reference. DataCenterId is the name by which the provider expects to be called when receiving an order, which could be different from the ProviderId.

(Note: When ECHO was first conceived, some providers had two systems. For instance, GSFC had both an ECS and a V0 system. To distinguish, ECHO called them GSFC-ECS and GSFC-V0. However, the order message expected the name GSFC. So, the DataCenterId field is used so that order messages could be filled in appropriately.)

---

```
<complexType name="OrderId">
<sequence>
<element name="ProviderId" type="string"/>
<element name="OrderId" type="string"/>
<element name="TrackingId" type="string"/>
<element name="DataCenterId" type="string"/>
</sequence>
</complexType>
```

---

## Line Items

The Line Item represents the item to be ordered (collection or granule) and all supporting information describing exactly what is being ordered. The first field of a Line Item is the GranuleUR (granule universal reference), which is your unique identifier for the granule. This is your unique identifier, not ECHO's unique identifier. This is optional if the Line Item being ordered is a data set. The next field is the DataSetId. This is the provider's unique identifier for a collection. It will be filled in with the identifier of the collection to which the granule belongs if the order is for a granule, and the identifier of the collection being ordered if the order is for a collection. The Quantity field allows for an order to specify how many of an item should be ordered. If the order is for electronic delivery

(that is, FTP), then the quantity should always be one. If the order is for the delivery of pre-packaged media, then the quantity can be considered the number of media that are being requested. The AuthenticationKey field is set based on the client calling SetAuthenticationKey. (This field can be no longer than 16 characters.) Finally, the options specified by the user regarding this line item are passed in as OrderOptions.

---

```
<complexType name="ListOfLineItems">
  <sequence>
    <element name="lineItems" type="tns:LineItem"
maxOccurs="unbounded"/>
  </sequence>
</complexType>

<complexType name="LineItem">
  <sequence>
    <element name="GranuleUr" type="string" minOccurs="0"/>
    <element name="DataSetId" type="string"/>
    <element name="Quantity" type="integer"/>
    <element name="AuthenticationKey" type="string"
minOccurs="0"/>
    <element name="OrderOptions" type="options:OptionSelection"
minOccurs="0" maxOccurs="unbounded"/>
  </sequence>
</complexType>
```

---

## User Information

User Information conveys who is placing the order. It consists of the ECHO User ID and the associated addresses for the order. While only the Contact Address is required, it should be noted that some Data Providers require all three addresses to be completed in order to fulfill the order.

---

```
<complexType name="UserInformation">
  <sequence>
    <element name="EchoUserId" type="string"/>
    <element name="ShippingAddress"
      type="tns:ShippingAddress" minOccurs="0"/>
    <element name="BillingAddress"
      type="tns:BillingAddress" minOccurs="0"/>
    <element name="ContactAddress"
      type="tns:ContactAddress"/>
  </sequence>
</complexType>
```

---

## Shipping Address

Use the **Shipping Address** to indicate where an order should be physically shipped (nonapplicable if you are using FTP for order fulfillment). The Name field contains the first and last name of the person to whom the order should be shipped. The Business Name contains the name of the company to whom the order should be shipped; this is an optional field. The Contact Name is the name of the person to be contacted if there are

---

any questions about shipping. The Phone Number contains a string with the telephone number of the contact name in case there are shipping issues. The Special Instruction field allows for more information for the delivery (for example, “Leave the package in front of the door.”) Finally, the Address field is the destination of the delivery.

---

```
<complexType name="ShippingAddress">
<sequence>
<element name="Name" type="tns:Name"/>
<element name="BusinessName" type="string" minOccurs="0"/>
<element name="ContactName" type="string"/>
<element name="PhoneNumber" type="string"/>
<element name="SpecialInstruction" type="string"/>
<element name="EmailAddress" type="string"/>
<element name="Address" type="tns:Address"/>
</sequence>
</complexType>
```

---

## Billing Address

The Billing Address is similar to the Shipping Address, and identifies who will be paying the bill for the order. There is no Special Instruction field for this case.

---

```
<complexType name="BillingAddress">
<sequence>
<element name="Name" type="tns:Name"/>
<element name="BusinessName" type="string" minOccurs="0"/>
<element name="ContactName" type="string"/>
<element name="PhoneNumber" type="string"/>
<element name="EmailAddress" type="string"/>
<element name="Address" type="tns:Address"/>
</sequence>
</complexType>
```

---

## Contact Address

The Contact Address is also similar, but it identifies who is placing the order.

---

```
<complexType name="ContactAddress">
<sequence>
<element name="Name" type="tns:Name"/>
<element name="PhoneNumber" type="string"/>
<element name="EmailAddress" type="string"/>
<element name="BusinessName" type="string"/>
<element name="Address" type="tns:Address"/>
</sequence>
</complexType>
```

---

## Name

This structure is simply a way to distinguish the first and last name of a person. You can append a middle initial or middle name to First Name if needed. You can also add a suffix to Last Name, such as Jr.

---

```
<complexType name="Name">
  <sequence>
    <element name="FirstName" type="string"/>
    <element name="LastName" type="string"/>
  </sequence>
</complexType>
```

---

## Address

The address field is a structure that describes an address. The ID field is a name that the user uses to describe the address (for example. Home, Work, Project1, etc.). The US Format flag is used to indicate whether a provider can rely on the address looking like a standard US address. The rule is that the system will validate those addresses that have the US Format flag set by checking that Street1, City, State, Zip Code, and Country are all set. If the US Format flag is clear, then only Street1 and Country are required. ECHO allows for up to 5 street address lines in either case.

---

```
<complexType name="Address">
  <sequence>
    <element name="Id" type="string"/>
    <element name="USFormat" type="boolean"/>
    <element name="Street1" type="string"/>
    <element name="Street2" type="string" minOccurs="0"/>
    <element name="Street3" type="string" minOccurs="0"/>
    <element name="Street4" type="string" minOccurs="0"/>
    <element name="Street5" type="string" minOccurs="0"/>
    <element name="City" type="string" minOccurs="0"/>
    <element name="State" type="political:State" minOccurs="0"/>
    <element name="ZipCode" type="string" minOccurs="0"/>
    <element name="Country" type="political:Country"/>
  </sequence>
</complexType>
```

---

## Client Identity

The client identity is unique for each ECHO client. Orders coming from an ECHO Client Partner carry this unique ID. Use the client identity to determine the source of any given order. This information is sent along with the other order information to the provider via the provider proxy. The client identity takes the form of a string generated by the client and is expected to look like “Client Name v1.0.”

*Note: Client identities are less than 16 characters.*

---

## Appendix A: Acronyms Used in ECHO

You will find the following acronyms frequently used in discussions of ECHO. This list is regularly updated at [http://www.echo.nasa.gov/overview/over\\_acronyms.shtml](http://www.echo.nasa.gov/overview/over_acronyms.shtml).

**ACL** - Access Control List

**API** - Application Programming Interface

**AQL** - Alternative Query Language

**ASF DAAC** - Alaska Satellite Facility DAAC

**ASTER** - Advanced Spaceborne Thermal Emission and Reflection Radiometer

**BMGT** - Bulk Metadata Generation Tool

**COTS** - Commercial Off The Shelf

**DAAC** - Distributed Active Archive Center

**DB** - DataBase

**DTD** - Document Type Definition

**ECHO** - EOS Clearinghouse

**ECS** - EOSDIS Core System

**EDC** - EROS Data Center

**EDG** - EOS Data Gateway

**EJB** - Enterprise JAVA Beans

**EMD** - EOSDIS Maintenance and Development

**EOS** - Earth Observing System

**EOSDIS** - EOS Data and Information System

**EROS** - Earth Resources Observation Systems

**ESDIS** - Earth Science Data and Information System

**ESIP** - Earth Science Information Partner

**ETC** - ECHO Technical Committee

**FTP** - File Transfer Protocol

**GCMD** - Global Change Master Directory

**GES DAAC** - GSFC Earth Sciences DAAC

**GHRC** - Global Hydrology Resource Center

**GIS** - Geographic Information System

**GML** - Geography Markup Language

**GMT** - Greenwich Mean Time

**GSFC** - Goddard Space Flight Center

**GUI** - Graphical User Interface

**GUID** - Globally Unique Identifier  
**IIMS** - Independent Information Management Subsystem  
**J2EE** - Java 2 Enterprise Edition  
**LAADS** - Level 1 and Atmosphere Archive and Distribution System  
**LP DAAC** - Land Processes DAAC  
**MISR** - Multiangle Imaging SpectroRadiometer  
**MODIS** - Moderate Resolution Imaging Spectroradiometer  
**NASA** - National Aeronautics and Space Administration  
**NSIDC DAAC** - National Snow and Ice Data Center DAAC  
**ODL** - Object Description Language  
**OGC** - OpenGIS Consortium  
**ORNL DAAC** - Oak Ridge National Laboratory DAAC  
**PGE** – Product Generation Executives  
**PO.DAAC** - Physical Oceanography DAAC  
**PSA** - Product Specific Attribute  
**PUMP** - Provider User Management Program  
**QA** - Quality Assurance  
**SEDAC** - Socioeconomic Data and Applications Center  
**SOAP** - Simple Object Access Protocol  
**SSC** - Stennis Space Center  
**SSL** - Secure Sockets Layer  
**UDDI** - Universal Description, Discovery and Integration  
**UI** - User Interface  
**UR** - Universal Reference  
**URI** – Uniform Resource Identifier  
**URL** - Uniform Resource Locator  
**UTC** - Universal Time, Coordinated (also called GMT/UTC)  
**WIST** - Warehouse Inventory Search Tool  
**WGS** – World Geodetic System  
**WRS** - Worldwide Reference System  
**WSDL** - Web Services Description Language  
**XML** - eXtensible Markup Language  
**XSLT** - eXtensible Style Language Transformation

## Appendix B: ECHO Path URIs

ECHO Path URI is the name given to a specific URI which maps into the metadata of a granule or collection.

### Format

The general format for the URI is

`<echoItemType>://<echoHost>/<echoItemId>[/xpath]`

- **echoItemType** := granule | collection . This indicates whether the URI is pointing to a granule or collections metadata.
- **echoHost** is the address of the ECHO server the metadata is on. This should be left blank for now as this feature is not supported.
- **echoItemId** is the item id of the of the granule or collection (for example, C14016455-PSATEST).
- **xpath** is optional. It is an XPath statement that maps into the XML payload returned from a GetMetadata request using the echoItemId.

Here is an example in the correct format. (Notice the echo host is left blank.)

- collection:///C14016455-PSATEST/%2Fresults%2Fprovider%2Fresult%2FCollectionMetaData%2FECHOItemId%2Ftext%28%29
- The XPath above is escaped to be put in the URI. The un-escaped format looks like: “/results/provider/result/CollectionMetaData/ECHOItemId/text()”

### Behavior

The granule mapping URI will retrieve data from the XML metadata of a granule or collection. The behavior that appears depending upon the state of the XPath selection appears below.

XPath Selection	Granule	Collection
No XPath included	Returns the entire metadata XML	Returns the entire metadata XML
XPath selects multiple nodes	Returns XML fragment representing multiple nodes	Returns XML fragment representing multiple nodes
XPath selects a node with child nodes and attributes	Returns XML fragment represent node and child nodes	Returns XML fragment represent node and child nodes
XPath selects a node with a single value	Returns single value	Returns single value

**This page is intentionally left blank.**

## Appendix C: ECHO Error Handling

The ECHO 9.0 Web Service API has advanced error-reporting capabilities. There are 12 types of faults reported by ECHO. They are:

- **AuthorizationFault** – Reported by ECHO when a user is not authorized to invoke an operation
- **DataSizeLimitFault** – Reported by ECHO to indicate that the data size limit has been exceeded
- **DuplicateIdFault** – Reported by ECHO to indicate that an entity with the same ID exists in ECHO already
- **InternalFault** – Reported by ECHO when an internal error occurs
- **InvalidArgumentException** – Reported to indicate that one or more arguments passed were invalid
- **InvalidStateFault** – Reported to indicate that an action by the client would put an object in ECHO in an invalid state
- **InvalidURLFault** – Reported to indicate invalid syntax in a URL or an element of the URL that does not exist
- **ItemNotFoundFault** – Reported when the client attempts to access one or more objects that do not exist
- **ParseFault** – Reported to indicate that some value could not be parsed
- **RemovalFault** - Reported to indicate an error that has occurred during the removal of an object from ECHO
- **UnsupportedFeatureFault** - Reported to indicated that a feature was selected that is not supported
- **ValidationFault** - Reported to indicate that an object in or passed to ECHO is not valid

All the above fault types extend the basic **EchoFault** type. All faults will include an **ErrorCode**, **SystemMessage**, **Timestamp** of when the error occurred and an **ErrorInstanceId**. An **EchoFault** may also have an **OpsMessage**.

Error codes are strings that uniquely identify an error case in ECHO. Some error codes are reused, such as when a required parameter to an operation was not provided.

Operations may associate different messages with specific error codes. If Operations has a message configured for an error code, then that message will be returned with the **EchoFault** in the **OpsMessage** element.

In most instances, receiving a fault from ECHO occurs by catching an **EchoFault** and displaying the Ops Message, System Message, and Error Instance ID to the user.

---

**Code Listing 56: Catching Exceptions from ECHO**

---

```
try
{
    // Create authentication service
    AuthenticationServiceLocator authServiceLocator =
        new AuthenticationServiceLocator();
    AuthenticationServicePort authenticationService =
        authServiceLocator.getAuthenticationServicePort();

    ClientInformation clientInfo =
        new ClientInformation();
    clientInfo.setClientId("A Client");
    clientInfo.setUserIpAddress("192.168.1.1");

    // Call login with jdoe as username, mypass as password,
    // and client information
    authenticationService.login("jdoe", "mypass",
        clientInfo, null, null);
}
catch (EchoFault e)
{
    // This exception was likely caused by user input.
    String message = "Could not login to ECHO:";

    if (e.getOpsMessage() != null
        && e.getOpsMessage().length() > 0)
    {
        message += "\nOps Message : " + e.getOpsMessage();
    }
    message +=
        "\nMessage: " + e.getSystemMessage()
            + "\nError Instance Id: "
            + e.getErrorInstanceId();
    System.out.println(message);
}
catch (RemoteException e)
{
    // ECHO could not be reached.
    System.out
        .println("Could not communicate with ECHO :"
            + e.toString());
}
catch (ServiceException e)
{
    // An error occurred while creating the service.
    System.out.println("Could not create ECHO Service :"
        + e.toString());
}
```

---

**InternalFaults** capture errors that are internal to ECHO, such as when ECHO cannot talk to the metadata catalog or when an order attempts to transition to an invalid state. There is nothing a client can do to recover from an **InternalFault**, except to report any information provided with the error to ECHO Operations.

**This page is intentionally left blank.**

## Appendix D: Ingest DTDs

The following DTDs can also be downloaded from the ECHO website at <http://www.echo.nasa.gov/reference/reference.shtml>.

### Collection Metadata DTD

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- edited with XML Spy v3.0b3 build 4 NT (http://www.xmlspy.com) by
Eva Tu (Global Science & Technology, Inc.) -->
<!-- Collection Metadata DTD for ECHO project - Release Version 1.4 -->
<!-- Release Date: April 2, 2001 -->
<!-- Modification History: Change the tag VersionId to VersionID to
reflect ECS's collection DTD modification -->
<!-- Modification History: Add in ParameterName for the ELEMENT
MediaParameters and -->
<!-- changed the ELEMENT MediaParameters from
single occurrence to multiple occurrence -->
<!-- Modification History: Took out SpatialType TAG that do not
associated with any other TAGs -->
<!-- Took out LAST_UPDATE TAG in Algorithm
Package Element that is not processed -->
<!-- Release Version 5.0 (version number to match ECHO system release)
-->
<!-- Added OnlineAccessURL, added OnlineResourceURL, removed Documents,
modified InnerRing -->
<!-- Release Date: Sep. 08, 2003 -->
<!-- Release Version 5.5 -->
<!-- Modified Spatial to support orbit and global and coordinate system
indication, Added Orderable, Added DataFormat -->
<!-- Modified Instrument to allow the association of
Collection/instrument/discipline Key word -->
<!-- Take out the TAGs of PrimaryCollectionFlag and ContainingGranules
-->
<!-- Release Date: Oct. 10, 2003 -->
<!-- Release Version 6.0 -->
<!-- Make Long name and Description required -->
<!-- Remove ELEMENT NewCollections and UpdateCollections -->
<!-- Re-define ELEMENT DeleteCollections to only require ShortName,
VersionID, and optional DeleteTime (DMR75) -->
<!-- Make mime type information optional - Jan. 12, 2004 -->
<!-- Make RangeEndingDate and RangeEndingTime optional - Jan. 12, 2004
-->
<!-- Take out DefaultPackage and CollectionPackage ELEMENTS - April 26,
2004 -->
<!-- Release Date: June 2004 -->
<!-- Release Version 7.0 -->
<!-- Remove AnalysisSource ELEMENT -->
<!-- Remove StorageMedium ELEMENT -->
<!-- Remove Review ELEMENT -->
<!-- Remove AlgorithmPackage associated ELEMENTs except name, version
and description -->
```

```

<!-- Remove ParentCollection ELEMENT -->
<!-- Release Date: -->
<!ELEMENT CollectionMetaDataFile (DTDVersion?, DataCenterId,
TemporalCoverage?, CollectionMetaDataSets)>
<!-- Version identifier of the DTD used to generate the file -->
<!ELEMENT DTDVersion (#PCDATA)>
<!-- DataCenterId of the site that stores this metadata (e.g., EDC-ECS)
-->
<!ELEMENT DataCenterId (#PCDATA)>
<!-- the start and end dates of this MetaDataFile (YYYYDDD) -->
<!ELEMENT TemporalCoverage (StartDate, EndDate)>
<!ELEMENT StartDate (#PCDATA)>
<!ELEMENT EndDate (#PCDATA)>
<!ELEMENT CollectionMetaDataSets (Collections?, DeleteCollections?)>
<!ELEMENT Collections (CollectionMetaData)*>
<!ELEMENT CollectionMetaData (ShortName, VersionID, InsertTime,
LastUpdate, DeleteTime?, LongName, DataSetID, CollectionDescription,
RevisionDate?, SuggestedUsage1?, SuggestedUsage2?, ProcessingCenter?,
ProcessingLevelId?, ProcessingLevelDescription?, ArchiveCenter?,
VersionDescription?, CitationforExternalPublication?, CollectionState?,
MaintenanceandUpdateFrequency?, RestrictionFlag?, RestrictionComment?,
Price?, Spatial?, Temporal?, Contact*, DisciplineTopicParameters*,
Platform*, AdditionalAttributes*, SpatialKeyword*, TemporalKeyword*,
CSDTDescription*, CollectionAssociation*, Campaign*, AlgorithmPackage*,
SpatialInfo*, OnlineAccessURLs?, CollectionOnlineResources?,
Orderable?, DataFormat?, AssociatedDIFs?)>
<!ELEMENT InsertTime (#PCDATA)>
<!ELEMENT LastUpdate (#PCDATA)>
<!ELEMENT DeleteTime (#PCDATA)>
<!ELEMENT ShortName (#PCDATA)>
<!ELEMENT LongName (#PCDATA)>
<!ELEMENT DataSetID (#PCDATA)>
<!ELEMENT CollectionDescription (#PCDATA)>
<!ELEMENT VersionID (#PCDATA)>
<!ELEMENT VersionDescription (#PCDATA)>
<!ELEMENT RevisionDate (#PCDATA)>
<!ELEMENT SuggestedUsage1 (#PCDATA)>
<!ELEMENT SuggestedUsage2 (#PCDATA)>
<!ELEMENT ProcessingCenter (#PCDATA)>
<!ELEMENT ArchiveCenter (#PCDATA)>
<!ELEMENT RestrictionFlag (#PCDATA)>
<!ELEMENT RestrictionComment (#PCDATA)>
<!ELEMENT CitationforExternalPublication (#PCDATA)>
<!ELEMENT CollectionState (#PCDATA)>
<!ELEMENT MaintenanceandUpdateFrequency (#PCDATA)>
<!ELEMENT ProcessingLevelId (#PCDATA)>
<!ELEMENT ProcessingLevelDescription (#PCDATA)>
<!ELEMENT Price (#PCDATA)>
<!ELEMENT Temporal (TimeType?, DateType?, TemporalRangeType?,
PrecisionofSeconds?, EndsatPresentFlag?, (RangeDateTime+ |
SingleDateTime+ | PeriodicDateTime+))>
<!ELEMENT TimeType (#PCDATA)>
<!ELEMENT DateType (#PCDATA)>
<!ELEMENT TemporalRangeType (#PCDATA)>
<!ELEMENT PrecisionofSeconds (#PCDATA)>
<!ELEMENT EndsatPresentFlag (#PCDATA)>

```

```

<!ELEMENT RangeDateTime (RangeBeginningDate, RangeBeginningTime,
RangeEndingDate?, RangeEndingTime?)>
<!ELEMENT RangeBeginningDate (#PCDATA)>
<!ELEMENT RangeBeginningTime (#PCDATA)>
<!ELEMENT RangeEndingDate (#PCDATA)>
<!ELEMENT RangeEndingTime (#PCDATA)>
<!ELEMENT SingleDateTime (CalendarDate, TimeOfDay)>
<!ELEMENT CalendarDate (#PCDATA)>
<!ELEMENT TimeOfDay (#PCDATA)>
<!ELEMENT PeriodicDateTime (PeriodName, Period1stDate, Period1stTime,
PeriodEndDate, PeriodEndTime, PeriodDurationUnit, PeriodDurationValue,
PeriodCycleDurationUnit, PeriodCycleDurationValue)>
<!ELEMENT PeriodName (#PCDATA)>
<!ELEMENT Period1stDate (#PCDATA)>
<!ELEMENT Period1stTime (#PCDATA)>
<!ELEMENT PeriodEndDate (#PCDATA)>
<!ELEMENT PeriodEndTime (#PCDATA)>
<!ELEMENT PeriodDurationUnit (#PCDATA)>
<!ELEMENT PeriodDurationValue (#PCDATA)>
<!ELEMENT PeriodCycleDurationUnit (#PCDATA)>
<!ELEMENT PeriodCycleDurationValue (#PCDATA)>
<!ELEMENT Spatial (SpatialCoverageType, HorizontalSpatialDomain?,
VerticalSpatialDomain*, OrbitParameters?, (GranuleSpatialRepresentation
| GranuleSpatialInheritance)?)>
<!ELEMENT SpatialCoverageType (#PCDATA)>
<!ELEMENT HorizontalSpatialDomain ((ZoneIdentifier?, Geometry) |
Global)>
<!ELEMENT ZoneIdentifier (#PCDATA)>
<!ELEMENT Global EMPTY>
<!ELEMENT Geometry (CoordinateSystem?, (Point | Circle |
BoundingRectangle | GPolygon | Polygon | Line)+)>
<!ELEMENT CoordinateSystem (Cartesian | Geodetic)>
<!ELEMENT Cartesian EMPTY>
<!ELEMENT Geodetic EMPTY>
<!ELEMENT Point (PointLongitude, PointLatitude)>
<!ELEMENT PointLongitude (#PCDATA)>
<!ELEMENT PointLatitude (#PCDATA)>
<!ELEMENT Circle (CenterLatitude, CenterLongitude, Radius)>
<!ELEMENT CenterLatitude (#PCDATA)>
<!ELEMENT CenterLongitude (#PCDATA)>
<!ELEMENT Radius (#PCDATA)>
<!ELEMENT BoundingRectangle (WestBoundingCoordinate,
NorthBoundingCoordinate, EastBoundingCoordinate,
SouthBoundingCoordinate)>
<!ELEMENT WestBoundingCoordinate (#PCDATA)>
<!ELEMENT NorthBoundingCoordinate (#PCDATA)>
<!ELEMENT EastBoundingCoordinate (#PCDATA)>
<!ELEMENT SouthBoundingCoordinate (#PCDATA)>
<!ELEMENT GPolygon (Boundary, ExclusiveZone?)>
<!ELEMENT ExclusiveZone (Boundary)+>
<!ELEMENT Boundary (Point, Point, Point, Point*)>
<!ELEMENT Polygon (SinglePolygon | MultiPolygon)>
<!ELEMENT SinglePolygon (OutRing, InnerRing?)>
<!ELEMENT OutRing (Boundary)>
<!ELEMENT InnerRing (Boundary)+>
<!ELEMENT MultiPolygon (SinglePolygon)+>

```

```

<!ELEMENT Line (Point, Point+)>
<!ELEMENT VerticalSpatialDomain (VerticalSpatialDomainType,
VerticalSpatialDomainValue)>
<!ELEMENT VerticalSpatialDomainType (#PCDATA)>
<!ELEMENT VerticalSpatialDomainValue (#PCDATA)>
<!ELEMENT OrbitParameters (SwathWidth, Period, InclinationAngle)>
<!ELEMENT SwathWidth (#PCDATA)>
<!ELEMENT Period (#PCDATA)>
<!ELEMENT InclinationAngle (#PCDATA)>
<!ELEMENT GranuleSpatialRepresentation (Cartesian | Geodetic | Orbit |
NoSpatial)>
<!ELEMENT Orbit EMPTY>
<!ELEMENT NoSpatial EMPTY>
<!ELEMENT GranuleSpatialInheritance EMPTY>
<!ELEMENT AdditionalAttributes (AdditionalAttributeDataType,
AdditionalAttributeDescription, AdditionalAttributeName,
MeasurementResolution?, ParameterRangeBegin?, ParameterRangeEnd?,
ParameterUnitsOfMeasure?, ParameterValueAccuracy?,
ValueAccuracyExplanation?, AdditionalAttributeValue?)>
<!ELEMENT AdditionalAttributeDataType (#PCDATA)>
<!ELEMENT AdditionalAttributeDescription (#PCDATA)>
<!ELEMENT AdditionalAttributeName (#PCDATA)>
<!ELEMENT MeasurementResolution (#PCDATA)>
<!ELEMENT ParameterRangeBegin (#PCDATA)>
<!ELEMENT ParameterRangeEnd (#PCDATA)>
<!ELEMENT ParameterUnitsOfMeasure (#PCDATA)>
<!ELEMENT ParameterValueAccuracy (#PCDATA)>
<!ELEMENT ValueAccuracyExplanation (#PCDATA)>
<!ELEMENT AdditionalAttributeValue (#PCDATA)>
<!ELEMENT CollectionAssociation (AssociatedShortName,
AssociatedVersionId, CollectionType, CollectionUse1?, CollectionUse2?)>
<!ELEMENT AssociatedShortName (#PCDATA)>
<!ELEMENT AssociatedVersionId (#PCDATA)>
<!ELEMENT CollectionType (#PCDATA)>
<!ELEMENT CollectionUse1 (#PCDATA)>
<!ELEMENT CollectionUse2 (#PCDATA)>
<!ELEMENT CSDTDescription (PrimaryCSDT, Implementation?, CSDTComments?,
IndirectReference?)>
<!ELEMENT PrimaryCSDT (#PCDATA)>
<!ELEMENT Implementation (#PCDATA)>
<!ELEMENT CSDTComments (#PCDATA)>
<!ELEMENT IndirectReference (#PCDATA)>
<!ELEMENT Contact (Role, HoursOfService?, ContactInstructions?,
ContactOrganizationName?, ContactOrganizationAddress*,
OrganizationTelephone*, OrganizationEmail*, ContactPersons*)>
<!ELEMENT Role (#PCDATA)>
<!ELEMENT HoursOfService (#PCDATA)>
<!ELEMENT ContactInstructions (#PCDATA)>
<!ELEMENT ContactOrganizationName (#PCDATA)>
<!ELEMENT ContactOrganizationAddress (StreetAddress, City,
StateProvince, PostalCode, Country)>
<!ELEMENT StreetAddress (#PCDATA)>
<!ELEMENT City (#PCDATA)>
<!ELEMENT StateProvince (#PCDATA)>
<!ELEMENT PostalCode (#PCDATA)>
<!ELEMENT Country (#PCDATA)>

```

```

<!ELEMENT OrganizationTelephone (TelephoneNumber, TelephoneType)>
<!ELEMENT TelephoneNumber (#PCDATA)>
<!ELEMENT TelephoneType (#PCDATA)>
<!ELEMENT OrganizationEmail (ElectronicEmailAddress)>
<!ELEMENT ElectronicEmailAddress (#PCDATA)>
<!ELEMENT ContactPersons (ContactFirstName, ContactMiddleName?,
ContactLastName, ContactJobPosition?)>
<!ELEMENT ContactFirstName (#PCDATA)>
<!ELEMENT ContactMiddleName (#PCDATA)>
<!ELEMENT ContactLastName (#PCDATA)>
<!ELEMENT ContactJobPosition (#PCDATA)>
<!ELEMENT SpatialKeyword (#PCDATA)>
<!ELEMENT TemporalKeyword (#PCDATA)>
<!ELEMENT Campaign (CampaignShortName, CampaignLongName?,,
CampaignStartDate?, CampaignEndDate?)>
<!ELEMENT CampaignShortName (#PCDATA)>
<!ELEMENT CampaignLongName (#PCDATA)>
<!ELEMENT CampaignStartDate (#PCDATA)>
<!ELEMENT CampaignEndDate (#PCDATA)>
<!ELEMENT DisciplineTopicParameters (DisciplineKeyword, TopicKeyword,
TermKeyword, VariableKeyword?, ECSParameterKeyword*)>
<!ELEMENT DisciplineKeyword (#PCDATA)>
<!ELEMENT TopicKeyword (#PCDATA)>
<!ELEMENT TermKeyword (#PCDATA)>
<!ELEMENT VariableKeyword (#PCDATA)>
<!ELEMENT ECSParameterKeyword (#PCDATA)>
<!ELEMENT Platform (PlatformShortName, PlatformLongName, PlatformType,
PlatformCharacteristic*, Instrument*)>
<!ELEMENT PlatformShortName (#PCDATA)>
<!ELEMENT PlatformLongName (#PCDATA)>
<!ELEMENT PlatformType (#PCDATA)>
<!ELEMENT PlatformCharacteristic (PlatformCharacteristicName,
PlatformCharacteristicDescription, PlatformCharacteristicDataType,
PlatformCharacteristicUnit, PlatformCharacteristicValue)>
<!ELEMENT PlatformCharacteristicName (#PCDATA)>
<!ELEMENT PlatformCharacteristicDescription (#PCDATA)>
<!ELEMENT PlatformCharacteristicDataType (#PCDATA)>
<!ELEMENT PlatformCharacteristicUnit (#PCDATA)>
<!ELEMENT PlatformCharacteristicValue (#PCDATA)>
<!ELEMENT Instrument (InstrumentShortName, InstrumentLongName?,
InstrumentTechnique?, NumberOfSensors?, InstrumentCharacteristic*,
Sensor*, OperationMode*, DisciplineTopicParameters*)>
<!ELEMENT InstrumentShortName (#PCDATA)>
<!ELEMENT InstrumentLongName (#PCDATA)>
<!ELEMENT InstrumentTechnique (#PCDATA)>
<!ELEMENT NumberOfSensors (#PCDATA)>
<!ELEMENT OperationMode (#PCDATA)>
<!ELEMENT InstrumentCharacteristic (InstrumentCharacteristicName,
InstrumentCharacteristicDescription, InstrumentCharacteristicDataType,
InstrumentCharacteristicUnit, InstrumentCharacteristicValue)>
<!ELEMENT InstrumentCharacteristicName (#PCDATA)>
<!ELEMENT InstrumentCharacteristicDescription (#PCDATA)>
<!ELEMENT InstrumentCharacteristicDataType (#PCDATA)>
<!ELEMENT InstrumentCharacteristicUnit (#PCDATA)>
<!ELEMENT InstrumentCharacteristicValue (#PCDATA)>

```

```

<!ELEMENT Sensor (SensorShortName, SensorLongName?, SensorTechnique?,
SensorCharacteristic*)>
<!ELEMENT SensorShortName (#PCDATA)>
<!ELEMENT SensorLongName (#PCDATA)>
<!ELEMENT SensorTechnique (#PCDATA)>
<!ELEMENT SensorCharacteristic (SensorCharacteristicName,
SensorCharacteristicDescription?, SensorCharacteristicDataType?,
SensorCharacteristicUnit?, SensorCharacteristicValue?)>
<!ELEMENT SensorCharacteristicName (#PCDATA)>
<!ELEMENT SensorCharacteristicDescription (#PCDATA)>
<!ELEMENT SensorCharacteristicDataType (#PCDATA)>
<!ELEMENT SensorCharacteristicUnit (#PCDATA)>
<!ELEMENT SensorCharacteristicValue (#PCDATA)>
<!ELEMENT AlgorithmPackage (AlgorithmPackageName,
AlgorithmPackageVersion, AlgorithmPackageDescription?)>
<!ELEMENT AlgorithmPackageName (#PCDATA)>
<!ELEMENT AlgorithmPackageVersion (#PCDATA)>
<!ELEMENT AlgorithmPackageDescription (#PCDATA)>
<!ELEMENT SpatialInfo (SpatialCoverage_Type, AltitudeDatumName?,
AltitudeDistanceUnits?, AltitudeEncodingMethod?, DepthDatumName?,
DepthDistanceUnits?, DepthEncodingMethod?, DenominatorofFlatteningRatio?,
EllipsoidName?, HorizontalDatumName?, SemiMajorAxis?, GeographicCoordinateUnits?,
LatitudeResolution?, LongitudeResolution?, LocalCoordinateSystemDesc?,
LocalGeoReferenceInformation?, PlanarCoordinateSystem*, DepthResolution*, AltitudeResolution*)>
<!ELEMENT SpatialCoverage_Type (#PCDATA)>
<!ELEMENT AltitudeDatumName (#PCDATA)>
<!ELEMENT AltitudeDistanceUnits (#PCDATA)>
<!ELEMENT AltitudeEncodingMethod (#PCDATA)>
<!ELEMENT DepthDatumName (#PCDATA)>
<!ELEMENT DepthDistanceUnits (#PCDATA)>
<!ELEMENT DepthEncodingMethod (#PCDATA)>
<!ELEMENT DenominatorofFlatteningRatio (#PCDATA)>
<!ELEMENT EllipsoidName (#PCDATA)>
<!ELEMENT HorizontalDatumName (#PCDATA)>
<!ELEMENT SemiMajorAxis (#PCDATA)>
<!ELEMENT GeographicCoordinateUnits (#PCDATA)>
<!ELEMENT LatitudeResolution (#PCDATA)>
<!ELEMENT LongitudeResolution (#PCDATA)>
<!ELEMENT LocalCoordinateSystemDesc (#PCDATA)>
<!ELEMENT LocalGeoReferenceInformation (#PCDATA)>
<!ELEMENT PlanarCoordinateSystem (PlanarCoordinateSystemID,
PlanarCoordinateEncodingMet?, PlanarDistanceUnits?,
BearingReferenceDirection?, BearingReferenceMeridian?,
BearingResolution?, BearingUnits?, DistanceResolution?,
AbscissaResolution?, OrdinateResolution?, MapProjectionName?,
MapProjectionPointer?, LocalPlanarCoordinateSystem?,
LocalPlanarGeoReferenceInfo?, GridCoordinateSystemName?)>
<!ELEMENT PlanarCoordinateSystemID (#PCDATA)>
<!ELEMENT PlanarCoordinateEncodingMet (#PCDATA)>
<!ELEMENT PlanarDistanceUnits (#PCDATA)>
<!ELEMENT BearingReferenceDirection (#PCDATA)>
<!ELEMENT BearingReferenceMeridian (#PCDATA)>
<!ELEMENT BearingResolution (#PCDATA)>
<!ELEMENT BearingUnits (#PCDATA)>

```

```
<!ELEMENT DistanceResolution (#PCDATA)>
<!ELEMENT AbscissaResolution (#PCDATA)>
<!ELEMENT OrdinateResolution (#PCDATA)>
<!ELEMENT MapProjectionName (#PCDATA)>
<!ELEMENT MapProjectionPointer (#PCDATA)>
<!ELEMENT LocalPlanarCoordinateSystem (#PCDATA)>
<!ELEMENT LocalPlanarGeoReferenceInfo (#PCDATA)>
<!ELEMENT GridCoordinateSystemName (#PCDATA)>
<!ELEMENT DepthResolution (#PCDATA)>
<!ELEMENT AltitudeResolution (#PCDATA)>
<!ELEMENT OnlineAccessURLs (OnlineAccessURL+)>
<!ELEMENT OnlineAccessURL (URL, URLDescription?,MimeType?)>
<!ELEMENT CollectionOnlineResources (OnlineResource+)>
<!ELEMENT OnlineResource (OnlineResourceURL,
OnlineResourceDescription?, OnlineResourceType,
OnlineResourceMimeType?)>
<!ELEMENT URL (#PCDATA)>
<!ELEMENT URLDescription (#PCDATA)>
<!ELEMENT MimeType (#PCDATA)>
<!ELEMENT OnlineResourceURL (#PCDATA)>
<!ELEMENT OnlineResourceDescription (#PCDATA)>
<!ELEMENT OnlineResourceType (#PCDATA)>
<!ELEMENT OnlineResourceMimeType (#PCDATA)>
<!ELEMENT Orderable EMPTY>
<!ELEMENT DataFormat (#PCDATA)>
<!ELEMENT DeleteCollections (DeleteCollection)*>
<!ELEMENT DeleteCollection (ShortName, VersionID, DeleteTime?)>
<!ELEMENT AssociatedDIFs (DIF+)>
<!ELEMENT DIF (EntryID)>
<!ELEMENT EntryID (#PCDATA)>
```

## Granule Metadata

```
<?xml version="1.0" encoding="US-ASCII"?>
<!-- edited with XML Spy v3.0.7 NT (http://www.xmlspy.com) by Eva Tu
(Global Science & Technology, Inc.) -->
<!-- Granule Metadata DTD for ECHO project - Release Version 1.0 -->
<!-- Release Date: November 28, 2000 -->
<!-- Granule Metadata DTD for ECHO project - Release Version 1.1 -->
<!-- Added AccessConstraint TAG -->
<!-- Release Date: May 16, 2001 -->
<!-- Release Version 5.0 (version number to match ECHO system release)
-->
<!-- Modified OnlineAccessURL, added OnlineResourceURL, removed
PSADefinition, modified InnerRing -->
<!-- Release Date: Sep. 08, 2003 -->
<!-- Release Version 5.5 -->
<!-- Modified SpatialDomainContainer to support orbit and global, Added
Orderable, Added DataFormat, Removed ECSDataGranule -->
<!-- Release Date: Oct. 10, 2003 -->
<!-- Release Version 6.0 -->
<!-- Removed ELEMENT NewGranules and UpdateGranules -->
<!-- Re-defined ELEMENT DeleteGranule to only require GranuleUR and
optional DeleteTime (DMR75) -->
<!-- Make mime type information optional - Jan. 12, 2004 -->
<!-- Remove ShortName ELEMENT for GranuleURMetaDate -->
<!-- Release Date: June 2004 -->
<!-- Release Version 7.0 -->
<!-- Remove AnalysisSource ELEMENT -->
<!-- Remove StorageMediumClass ELEMENT -->
<!-- Remove FileStorage ELEMENT -->
<!-- Remove Review ELEMENT -->
<!-- Remove ProcessingQA ELEMENT -->
<!-- Remove QAProduct ELEMENT -->
<!-- Remove PHPProduct ELEMENT -->
<!-- Remove VersionHistory ELEMENT -->
<!-- Remove PlatformCharacteristic ELEMENT -->
<!-- Remove Platform associated ELEMENTs except PlatformShortName -->
<!-- Remove Instrument associated ELEMENT except InstrumentShortName
and InstrumentCharacteristic -->
<!-- Remove Sensor associated ELEMENTs except SensorShortName and
SensorCharacteristic -->
<!-- Remove Campaign associated ELEMENTs except CampaignShortName -->
<!-- Remove Contact ELEMENT -->
<!-- Remove DbID ELEMENT -->
<!-- Remove GranuleVersionID ELEMENT -->
<!-- Release Date: -->
<!ELEMENT GranuleMetaDataFile (DTDVersion?, DataCenterId,
TemporalCoverage?, GranuleMetaDataSet)>
<!-- Version identifier of the DTD used to generate the file -->
<!ELEMENT DTDVersion (#PCDATA)>
<!-- DataCenterId of the site that stores this metadata (e.g., EDC-ECS)
-->
<!ELEMENT DataCenterId (#PCDATA)>
```

```

<!-- the start and end dates of this MetaDataFile (YYYYDDD) -->
<!ELEMENT TemporalCoverage (StartDate, EndDate)>
<!ELEMENT StartDate (#PCDATA)>
<!ELEMENT EndDate (#PCDATA)>
<!ELEMENT GranuleMetaDataSet (Granules?, DeleteGranules?)>
<!ELEMENT Granules (GranuleURMetaData)+>
<!ELEMENT GranuleURMetaData (GranuleUR, InsertTime, LastUpdate,
DeleteTime?, CollectionMetaData, RestrictionFlag?, RestrictionComment?,
DataGranule?, PGEVersionClass?, (RangeDateTime | SingleDateTime)?,
SpatialDomainContainer?, OrbitCalculatedSpatialDomain?,
MeasuredParameter?, Platform*, Campaign*, (AdditionalAttributes |
PSAs)?, InputGranule?, TwoDCoordinateSystem?, Price?,
OnlineAccessURLs?, GranuleOnlineResources?, Orderable?, DataFormat?)>
<!ELEMENT Orderable EMPTY>
<!ELEMENT OnlineAccessURLs (OnlineAccessURL+)>
<!ELEMENT OnlineAccessURL (URL, URLDescription?,MimeType?)>
<!ELEMENT GranuleOnlineResources (OnlineResource+)>
<!ELEMENT OnlineResource (OnlineResourceURL,
OnlineResourceDescription?, OnlineResourceType,
OnlineResourceMimeType?)>
<!ELEMENT URL (#PCDATA)>
<!ELEMENT DataFormat (#PCDATA)>
<!ELEMENT URLDescription (#PCDATA)>
<!ELEMENT MimeType (#PCDATA)>
<!ELEMENT OnlineResourceURL (#PCDATA)>
<!ELEMENT OnlineResourceDescription (#PCDATA)>
<!ELEMENT OnlineResourceType (#PCDATA)>
<!ELEMENT OnlineResourceMimeType (#PCDATA)>
<!ELEMENT GranuleUR (#PCDATA)>
<!ELEMENT InsertTime (#PCDATA)>
<!ELEMENT LastUpdate (#PCDATA)>
<!ELEMENT DeleteTime (#PCDATA)>
<!ELEMENT CollectionMetaData (((ShortName, VersionID) | DatasetID),
PrimaryFlag?)>
<!ELEMENT ShortName (#PCDATA)>
<!ELEMENT VersionID (#PCDATA)>
<!ELEMENT DatasetID (#PCDATA)>
<!ELEMENT PrimaryFlag (#PCDATA)>
<!ELEMENT RestrictionFlag (#PCDATA)>
<!ELEMENT RestrictionComment (#PCDATA)>
<!ELEMENT DataGranule (SizeMBDataGranule?, ReprocessingPlanned?,
ReprocessingActual?, ProducerGranuleID?, DayNightFlag?,
ProductionDateTime, LocalVersionID?)>
<!ELEMENT SizeMBDataGranule (#PCDATA)>
<!ELEMENT ReprocessingPlanned (#PCDATA)>
<!ELEMENT ReprocessingActual (#PCDATA)>
<!ELEMENT ProducerGranuleID (#PCDATA)>
<!ELEMENT DayNightFlag (#PCDATA)>
<!ELEMENT ProductionDateTime (#PCDATA)>
<!ELEMENT LocalVersionID (#PCDATA)>
<!ELEMENT PGEVersionClass (PGEName?, PGEVersion)>
<!ELEMENT PGEVersion (#PCDATA)>
<!ELEMENT PGEName (#PCDATA)>
<!ELEMENT RangeDateTime (RangeEndingTime, RangeEndDate,
RangeBeginningTime, RangeBeginningDate)>
<!ELEMENT RangeEndingTime (#PCDATA)>

```

```

<!ELEMENT RangeEndingDate (#PCDATA)>
<!ELEMENT RangeBeginningTime (#PCDATA)>
<!ELEMENT RangeBeginningDate (#PCDATA)>
<!ELEMENT SingleDateTime (TimeOfDay, CalendarDate)>
<!ELEMENT TimeOfDay (#PCDATA)>
<!ELEMENT CalendarDate (#PCDATA)>
<!ELEMENT SpatialDomainContainer (GranuleLocality?,
VerticalSpatialDomain?, HorizontalSpatialDomainContainer?)>
<!ELEMENT GranuleLocality (LocalityValue)+>
<!ELEMENT LocalityValue (#PCDATA)>
<!ELEMENT VerticalSpatialDomain (VerticalSpatialDomainContainer)+>
<!ELEMENT VerticalSpatialDomainContainer (VerticalSpatialDomainType?,
VerticalSpatialDomainValue?)>
<!ELEMENT VerticalSpatialDomainType (#PCDATA)>
<!ELEMENT VerticalSpatialDomainValue (#PCDATA)>
<!ELEMENT HorizontalSpatialDomainContainer (ZoneIdentifier?, ((Point |
Circle | BoundingRectangle | GPolygon | Polygon | Line)+ | Orbit |
Global))>
<!ELEMENT ZoneIdentifier (#PCDATA)>
<!ELEMENT Circle (CenterLatitude, CenterLongitude, Radius)>
<!ELEMENT CenterLatitude (#PCDATA)>
<!ELEMENT CenterLongitude (#PCDATA)>
<!ELEMENT Radius (#PCDATA)>
<!ELEMENT BoundingRectangle (WestBoundingCoordinate,
NorthBoundingCoordinate, EastBoundingCoordinate,
SouthBoundingCoordinate, CenterPoint?)>
<!ELEMENT WestBoundingCoordinate (#PCDATA)>
<!ELEMENT NorthBoundingCoordinate (#PCDATA)>
<!ELEMENT EastBoundingCoordinate (#PCDATA)>
<!ELEMENT SouthBoundingCoordinate (#PCDATA)>
<!ELEMENT CenterPoint (CenterLatitude, CenterLongitude)>
<!ELEMENT GPolygon (Boundary, ExclusiveZone?, CenterPoint?)>
<!ELEMENT ExclusiveZone (Boundary)+>
<!ELEMENT Boundary (Point, Point, Point, Point*)>
<!ELEMENT Point (PointLongitude, PointLatitude)>
<!ELEMENT PointLongitude (#PCDATA)>
<!ELEMENT PointLatitude (#PCDATA)>
<!ELEMENT Polygon (SinglePolygon | MultiPolygon)>
<!ELEMENT SinglePolygon (OutRing, InnerRing?, CenterPoint?)>
<!ELEMENT OutRing (Boundary)>
<!ELEMENT InnerRing (Boundary)+>
<!ELEMENT MultiPolygon (SinglePolygon)+>
<!ELEMENT Line (Point, Point+, CenterPoint?)>
<!ELEMENT Orbit (AscendingCrossing, StartLat, StartDirection, EndLat,
EndDirection, NumberOfOrbit?, CenterPoint?)>
<!ELEMENT AscendingCrossing (#PCDATA)>
<!ELEMENT StartLat (#PCDATA)>
<!ELEMENT StartDirection (#PCDATA)>
<!ELEMENT EndLat (#PCDATA)>
<!ELEMENT EndDirection (#PCDATA)>
<!ELEMENT NumberOfOrbit (#PCDATA)>
<!ELEMENT Global EMPTY>
<!ELEMENT OrbitCalculatedSpatialDomain
(OrbitCalculatedSpatialDomainContainer)+>

```

```

<!ELEMENT OrbitCalculatedSpatialDomainContainer (OrbitalModelName?,
OrbitNumber?, StartOrbitNumber?, StopOrbitNumber?,
EquatorCrossingLongitude?, EquatorCrossingDate?, EquatorCrossingTime?)>
<!ELEMENT OrbitalmodelName (#PCDATA)>
<!ELEMENT OrbitNumber (#PCDATA)>
<!ELEMENT StartOrbitNumber (#PCDATA)>
<!ELEMENT StopOrbitNumber (#PCDATA)>
<!ELEMENT EquatorCrossingLongitude (#PCDATA)>
<!ELEMENT EquatorCrossingDate (#PCDATA)>
<!ELEMENT EquatorCrossingTime (#PCDATA)>
<!ELEMENT MeasuredParameter (MeasuredParameterContainer)+>
<!ELEMENT MeasuredParameterContainer (ParameterName, QAStats?,
QAFlags?)>
<!ELEMENT ParameterName (#PCDATA)>
<!ELEMENT QAStats (QAPercentMissingData?, QAPercentOutofBoundsData?,
QAPercentInterpolatedData?, QAPercentCloudCover?)>
<!ELEMENT QAPercentMissingData (#PCDATA)>
<!ELEMENT QAPercentOutofBoundsData (#PCDATA)>
<!ELEMENT QAPercentInterpolatedData (#PCDATA)>
<!ELEMENT QAPercentCloudCover (#PCDATA)>
<!ELEMENT QAFlags (AutomaticQualityFlag?,
AutomaticQualityFlagExplanation?, OperationalQualityFlag?,
OperationalQualityFlagExplanation?, ScienceQualityFlag?,
ScienceQualityFlagExplanation?)>
<!ELEMENT AutomaticQualityFlag (#PCDATA)>
<!ELEMENT AutomaticQualityFlagExplanation (#PCDATA)>
<!ELEMENT OperationalQualityFlag (#PCDATA)>
<!ELEMENT OperationalQualityFlagExplanation (#PCDATA)>
<!ELEMENT ScienceQualityFlag (#PCDATA)>
<!ELEMENT ScienceQualityFlagExplanation (#PCDATA)>
<!ELEMENT Platform (PlatformShortName, Instrument*)>
<!ELEMENT PlatformShortName (#PCDATA)>
<!ELEMENT Instrument (InstrumentShortName, InstrumentCharacteristic*, Sensor*, OperationMode*)>
<!ELEMENT InstrumentShortName (#PCDATA)>
<!ELEMENT OperationMode (#PCDATA)>
<!ELEMENT InstrumentCharacteristic (InstrumentCharacteristicName, InstrumentCharacteristicDescription?, InstrumentCharacteristicDataType?, InstrumentCharacteristicUnit?, InstrumentCharacteristicValue)>
<!ELEMENT InstrumentCharacteristicName (#PCDATA)>
<!ELEMENT InstrumentCharacteristicDescription (#PCDATA)>
<!ELEMENT InstrumentCharacteristicDataType (#PCDATA)>
<!ELEMENT InstrumentCharacteristicUnit (#PCDATA)>
<!ELEMENT InstrumentCharacteristicValue (#PCDATA)>
<!ELEMENT Sensor (SensorShortName, SensorCharacteristic*)>
<!ELEMENT SensorShortName (#PCDATA)>
<!ELEMENT SensorCharacteristic (SensorCharacteristicName, SensorCharacteristicDescription?, SensorCharacteristicDataType?, SensorCharacteristicUnit?, SensorCharacteristicValue)>
<!ELEMENT SensorCharacteristicName (#PCDATA)>
<!ELEMENT SensorCharacteristicDescription (#PCDATA)>
<!ELEMENT SensorCharacteristicDataType (#PCDATA)>
<!ELEMENT SensorCharacteristicUnit (#PCDATA)>
<!ELEMENT SensorCharacteristicValue (#PCDATA)>
<!ELEMENT Campaign (CampaignShortName)>

```

```
<!ELEMENT CampaignShortName (#PCDATA)>
<!ELEMENT PSAs (PSA)+>
<!ELEMENT PSA (PSAName, PSAValue+, PSAValueType?)>
<!ELEMENT PSAName (#PCDATA)>
<!ELEMENT PSAValue (#PCDATA)>
<!ELEMENT PSAValueType (#PCDATA)>
<!ELEMENT AdditionalAttributes (AdditionalAttribute)+>
<!ELEMENT AdditionalAttribute (AdditionalAttributeName,
AdditionalAttributeValue*, AdditionalAttributeValueType?)>
<!ELEMENT AdditionalAttributeName (#PCDATA)>
<!ELEMENT AdditionalAttributeValue (#PCDATA)>
<!ELEMENT AdditionalAttributeValueType (#PCDATA)>
<!ELEMENT InputGranule (InputPointer)+>
<!ELEMENT InputPointer (#PCDATA)>
<!ELEMENT TwoDCoordinateSystem (StartCoordinate1, EndCoordinate1?,
StartCoordinate2, EndCoordinate2?, TwoDCoordinateSystemName)>
<!ELEMENT StartCoordinate1 (#PCDATA)>
<!ELEMENT EndCoordinate1 (#PCDATA)>
<!ELEMENT StartCoordinate2 (#PCDATA)>
<!ELEMENT EndCoordinate2 (#PCDATA)>
<!ELEMENT TwoDCoordinateSystemName (#PCDATA)>
<!ELEMENT Price (#PCDATA)>
<!ELEMENT DeleteGranules (DeleteGranule)+>
<!ELEMENT DeleteGranule (GranuleUR, DeleteTime?)>
```

## Update Metadata DTD

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- edited with XML Spy v3.5 NT (http://www.xmlspy.com) by Keith
Wichmann (Global Science and Technology, Inc.) -->
<!--UpdateMetadata can update a single collection, multiple
collections, a single granule, or multiple granules in one transaction.
Each update allows the addition of new metadata-->
<!ELEMENT UpdateMetadata (Collection*, Granule*)>
<!ELEMENT Collection (Target+, (Add | Update | Delete)+)>
<!ELEMENT Granule (Target+, (Add | Update | Delete)+)>
<!-- Target+ allows the same change to be made to several different
granules or collections simultaneously. This is especially useful for
bulk deletions of OnlineURLs. -->
<!ELEMENT Target (ID, ProviderLastUpdateDateTime, SaveDateTimeFlag?)>
<!-- SaveDateTimeFlag is the flag that allows echo to update the last
update date time for Target. The default is SAVE -->
<!ELEMENT Add (QualifiedTag, MetadataValue)>
<!ELEMENT Update (QualifiedTag, MetadataValue)>
<!ELEMENT Delete (QualifiedTag+)>
<!ELEMENT QualifiedTag (#PCDATA)>
<!ELEMENT MetadataValue (#PCDATA)>
<!ELEMENT ProviderLastUpdateDateTime (#PCDATA)>
<!ELEMENT SaveDateTimeFlag (SAVE | DONTSAVE)>
<!ELEMENT SAVE EMPTY>
<!ELEMENT ID (#PCDATA)>
<!ELEMENT DONTSAVE EMPTY>
```

## Browse Metadata DTD

```
<?xml version="1.0" encoding="UTF-8"?>
<!--DTD generated by XMLSpy v2005 rel. 3 U (http://www.altova.com)-->
<!ELEMENT BrowseReferenceFile (DTDVersion, DataCenterId,
TemporalCoverage, DeleteBrowse*, BrowseCrossReference*)>
<!ELEMENT DTDVersion (#PCDATA)>
<!ELEMENT DataCenterId (#PCDATA)>
<!ELEMENT TemporalCoverage (StartDate, EndDate)>
<!ELEMENT StartDate (#PCDATA)>
<!ELEMENT EndDate (#PCDATA)>
<!ELEMENT DeleteBrowse (((ShortName, VersionID) | DataSetID |
GranuleUR), InternalFileName*, BrowseCollectionId?, BrowseGranuleId?)>
<!ATTLIST DeleteBrowse
    SilentDeleteErrorHandler NMTOKEN "0"
>
<!ELEMENT ShortName (#PCDATA)>
<!ELEMENT VersionID (#PCDATA)>
<!ELEMENT DataSetID (#PCDATA)>
<!ELEMENT GranuleUR (#PCDATA)>
<!ELEMENT InternalFileName (#PCDATA)>
<!ELEMENT BrowseCollectionId (#PCDATA)>
<!ELEMENT BrowseGranuleId (#PCDATA)>
<!ELEMENT BrowseCrossReference (((ShortName, VersionID) | DataSetID |
GranuleUR), BrowseCollectionId?, BrowseGranuleId?, InsertTime?,
LastUpdate?, DeleteTime?, InternalFileName, BrowseDescription?,
BrowseSize)>
<!ELEMENT InsertTime (#PCDATA)>
<!ELEMENT LastUpdate (#PCDATA)>
<!ELEMENT DeleteTime (#PCDATA)>
<!ELEMENT BrowseDescription (#PCDATA)>
<!ELEMENT BrowseSize (#PCDATA)
```

# **Appendix E: Best Practices for Preparing Your Metadata**

The following tips and other recommended practices are in outline form rather than paragraphs for quick reading.

## **Tips**

### **Submission of Large Numbers of Files and Granules**

- Organizing granule metadata in files, each containing 1000 granules, is ideal for fastest processing.
  - This minimizes file I/O and does not require the execution of a chunking process that occurs on larger files with more than 1000 granules.
  - Many metadata files with only a few records per file can slow ingest by an order of magnitude.
  - “Chunking” large files (slows ingest, but to a much lesser degree).

### **Tools to Help Prepare XML**

- Entity Definitions
- DTDs
- Examples of minimum requirements

### **Additional Tips**

- File Naming conventions.
- Validate before submission
- Perform ingest of new metadata on partner test system
- There are separate DTDs that govern the XML format for the various metadata types, available from the ECHO website: <http://www.echo.nasa.gov/>.

There are also examples of minimum collection and granule metadata XML requirements located on the ECHO website—click Data Partners, then Getting Started as a Data Partner.

Entity definitions exist that detail data type, character limitations, as well as definitions of the various elements of for each metadata type. These definitions can be found at <http://www.echo.nasa.gov/>; just click Data Partners, then ECHO Data Model, then Entity Info.

A metadata file name should be self-describing and unique, including ID of data provider, metadata type, short name and version\_id. The date coverage of metadata and generation date is helpful, for example:

NSCGAMSR200701620070160101.20070116113342.XML

EDCBMOLT200701620070170202.20070117005857.XML

- Perform your own validation before submitting metadata for ingest
- Perform a test ingest for all metadata types using the Partner Test System before attempting an operational ingest

## Factors That Affect Ingest Rates

- Organize browse metadata so that multiple browse imagery references are contained within one metadata file
- Metadata organization (many vs. few records per metadata file)
- Metadata type being ingested (collection, granule, browse, update)
- Type of action being taken (insert, delete, replacement)
- Proportion of metadata types in an ingest job
- Amount of information in a metadata record (sparse vs. dense)
- Number of records in a provider schema
- Amount of time required for preprocessing
- Amount of competition for database and system resources

## Common Errors

- Use of incorrect data type for element:
  - Use of string for a date or number field:

```
<DeleteTime>none</DeleteTime>
```

```
<SwathWidth>2600 km</SwathWidth>
```
- Exceeding the character limitations for an element—refer to entity definitions on ECHO website.
- Incomplete dates
- Not matching short name and version\_id of granule to what is used with its associated collection.
- Violating unique constraints within various tables in the provider schema (DataSetId, GranuleUR, etc)
- ECHO Ingest currently performs the following data integrity checks
  - Additional Attribute Name (Product Specific Attribute)
  - Platform/Instrument/Sensor configuration
  - Platform/Instrument/Instrument\_Operation\_Mode
  - Campaign
  - Analysis Source

(These must be defined at the collection level before using at the granule level, and the usage must be consistent.)

**This page is intentionally left blank.**

# Index

## B

Browse  
  delete example, 72  
  DTD, 68  
  element definitions, 69  
  image files, 66  
  ingest, 66  
  insert example, 70  
  metadata, 66  
  update example, 71

## C

Catalog  
  introduction to the term, 3  
Collection  
  introduction to the term, 4

## D

Data access rules  
  conditions, 83  
  group, 83  
  managing access, 83  
  PUMP, 83  
  restriction flag, 84  
DTD  
  browse, 68  
  code listing example for metadata update DTD, 58  
  data not included in the DTD, 59  
  for collection metadata, 59  
  location for all ECHO DTDs, 3  
  metadata updated DTD, 58  
  separate DTDs for updating granules, collections, 4

## E

ECHO  
  design, 2  
  entities, 8  
  features, 2  
  graphical representation of concept, 2  
Element  
  browse, 69  
Error handling  
  catching exceptions, 100  
  types of errors, 99  
Error message  
  data that violates data constraints, 78  
  example of granule associated with nonexistent collection, 79  
  example of return from invalid deletion, 78  
  granule associated with nonexistent collection, 79  
  invalid deletion, 78

wrong date format, 80  
Error messages, 73  
  duplicate input, 77  
  example of Ingest Summary Report, 74  
  example of input file validation error, 73  
  example of XML validation error, 74  
  input file validation error, 73  
  invalid XML file error, 74  
  non-XML, 73  
Exception catching, 100

## F

FTP, 3  
  ingest, 3  
  location for ingest, 13

## G

Granule  
  introduction to the term, 4  
GUID  
  defined, 8

## I

Ingest  
  additional attributes, 14  
  browse files, 66  
  FTP location, 13  
  metadata mapping, 14  
  overview, 13  
  product specific attributes, 14  
  types of ingest information, 75

## M

Metadata mapping  
  ingest, 14

## O

Order options  
  defined, 10

## V

Validating your metadata  
  viewing dataset information, 81

## X

XPath  
  values listed in a table, 61